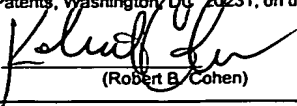
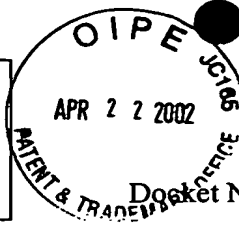


I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail, in an envelope addressed to: Commissioner for Patents, Washington, DC 20231, on the date shown below.

Dated: April 17, 2002 Signature: 

(Robert B. Cohen)



Docket No.: SONYAK 3.9-157 CIP  
(PATENT)

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Patent Application of:  
Kato et al.

Application No.: 10/029,903

Group Art Unit: 2851

Filed: December 21, 2001

Examiner: Not Yet Assigned

For: INFORMATION PROCESSING METHOD  
AND APPARATUS, PROGRAM AND  
RECORDING MEDIUM

**CLAIM FOR PRIORITY AND SUBMISSION OF DOCUMENTS**

Commissioner for Patents  
Washington, DC 20231

RECEIVED  
APR 24 2002  
TECHNOLOGY CENTER 2800

Dear Sir:

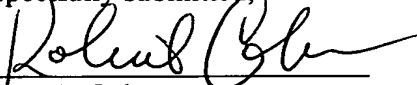
Applicant hereby claims priority under 35 U.S.C. 119 based on the following prior foreign applications filed in the following foreign countries on the dates indicated:

Country	Application No.	Date
Japan	2000-183769	April 21, 2000
Japan	2000-271550	September 7, 2000

In support of this claim, certified copies of the original foreign applications are filed herewith.

Dated: April 17, 2002

Respectfully submitted,

By 

Robert B. Cohen

Registration No.: 32,768  
LERNER, DAVID, LITTENBERG,  
KRUMHOLZ & MENTLIK, LLP  
600 South Avenue West  
Westfield, New Jersey 07090  
(908) 654-5000  
Attorneys for Applicant



日 本 国 特 許 庁  
JAPAN PATENT OFFICE

5

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2000年 4月21日

出 願 番 号

Application Number:

特願2000-183769

ST.10/C ]:

[JP2000-183769]

出 願 人

Applicant(s):

ソニー株式会社

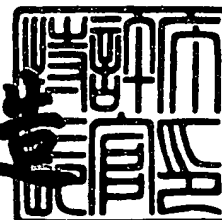
RECEIVED  
APR 24 2002  
TECHNOLOGY CENTER 2800

CERTIFIED COPY OF  
PRIORITY DOCUMENT

2002年 2月 1日

特 許 庁 長 官  
Commissioner,  
Japan Patent Office

及 川 耕 造



出証番号 出証特2002-3003666

【書類名】 特許願

【整理番号】 0000369403

【提出日】 平成12年 4月21日

【あて先】 特許庁長官殿

【国際特許分類】 H04N 5/76

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社  
内

【氏名】 加藤 元樹

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社  
内

【氏名】 浜田 俊也

【特許出願人】

【識別番号】 000002185

【氏名又は名称】 ソニー株式会社

【代表者】 出井 伸之

【代理人】

【識別番号】 100082131

【弁理士】

【氏名又は名称】 稲本 義雄

【電話番号】 03-3369-6479

【手数料の表示】

【予納台帳番号】 032089

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

特 2 0 0 0 - 1 8 3 7 6 9

【包括委任状番号】 9708842

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 情報処理装置および方法、並びに記録媒体

【特許請求の範囲】

【請求項 1】 第 1 の AV ストリームから第 2 の AV ストリームへ連続的に再生されるように指示された場合、前記第 1 の AV ストリームの所定の部分と前記第 2 の AV ストリームの所定の部分から構成され、前記第 1 の AV ストリームから前記第 2 の AV ストリームに再生が切り換えられるとき再生される第 3 の AV ストリームを生成する第 1 の生成手段と、

前記第 1 の生成手段により生成された前記第 3 の AV ストリームの情報として、前記第 1 の AV ストリームから前記第 3 の AV ストリームに再生が切り替わるとき再生される前記第 1 の AV ストリームのソースパケットのアドレスの情報と、前記第 3 の AV ストリームから前記第 2 の AV ストリームに再生が切り替わるとき再生される前記第 2 の AV ストリームのソースパケットのアドレスの情報から構成されるアドレス情報を生成する第 2 の生成手段と、

前記第 1 の生成手段により生成された前記第 3 の AV ストリームと、前記第 2 の生成手段により生成された前記アドレス情報を記録する記録手段と

を含むことを特徴とする情報処理装置。

【請求項 2】 前記第 2 の生成手段により生成された前記アドレス情報に含まれる前記第 1 の AV ストリームのソースパケットのアライバルタイムスタンプと、前記第 3 の AV ストリームの最初に位置するソースパケットのアライバルタイムスタンプは連続しており、かつ、前記第 2 の生成手段により生成された前記アドレス情報に含まれる前記第 2 の AV ストリームのソースパケットのアライバルタイムスタンプと、前記第 3 の AV ストリームの最後に位置するソースパケットのアライバルタイムスタンプは連続している

ことを特徴とする請求項 1 に記載の情報処理装置。

【請求項 3】 前記第 3 の AV ストリーム内のソースパケットのアライバルタイムスタンプには、ただ 1 つの不連続点が存在する

ことを特徴とする請求項 2 に記載の情報処理装置。

【請求項 4】 第 1 の AV ストリームから第 2 の AV ストリームへ連続的に再生

されるように指示された場合、前記第 1 の AV ストリームの所定の部分と前記第 2 の AV ストリームの所定の部分から構成され、前記第 1 の AV ストリームから前記第 2 の AV ストリームに再生が切り換えられるとき再生される第 3 の AV ストリームを生成する第 1 の生成ステップと、

前記第 1 の生成ステップの処理で生成された前記第 3 の AV ストリームの情報として、前記第 1 の AV ストリームから前記第 3 の AV ストリームに再生が切り替わるとき再生される前記第 1 の AV ストリームのソースパケットのアドレスの情報と、前記第 3 の AV ストリームから前記第 2 の AV ストリームに再生が切り替わる時再生される前記第 2 の AV ストリームのソースパケットのアドレスの情報から構成されるアドレス情報を生成する第 2 の生成ステップと

を含むことを特徴とする情報処理方法。

【請求項 5】 第 1 の AV ストリームから第 2 の AV ストリームへ連続的に再生されるように指示された場合、前記第 1 の AV ストリームの所定の部分と前記第 2 の AV ストリームの所定の部分から構成され、前記第 1 の AV ストリームから前記第 2 の AV ストリームに再生が切り換えられるとき再生される第 3 の AV ストリームを生成する第 1 の生成ステップと、

前記第 1 の生成ステップの処理で生成された前記第 3 の AV ストリームの情報として、前記第 1 の AV ストリームから前記第 3 の AV ストリームに再生が切り替わる時再生される前記第 1 の AV ストリームのソースパケットのアドレスの情報と、前記第 3 の AV ストリームから前記第 2 の AV ストリームに再生が切り替わる時再生される前記第 2 の AV ストリームのソースパケットのアドレスの情報から構成されるアドレス情報を生成する第 2 の生成ステップと

を含むことを特徴とするコンピュータが読み取り可能なプログラムが記録されている記録媒体。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は情報処理装置および方法、並びに記録媒体に関し、特に、再生区間おける動画像の連続性を保つ情報処理装置および方法、並びに記録媒体に関する。

## 【 0 0 0 2 】

## 【従来の技術】

近年、記録再生装置から取り外し可能なディスク型の記録媒体として、各種の光ディスクが提案されつつある。このような記録可能な光ディスクは、数ギガバイトの大容量メディアとして提案されており、ビデオ信号等のAV(Audio Visual)信号を記録するメディアとしての期待が高い。この記録可能な光ディスクに記録するデジタルのAV信号のソース（供給源）としては、CSデジタル衛星放送やBSデジタル放送があり、また、将来はデジタル方式の地上波テレビジョン放送等も提案されている。

## 【 0 0 0 3 】

ここで、これらのソースから供給されるデジタルビデオ信号は、通常MPEG (Moving Picture Experts Group) 2 方式で画像圧縮されているのが一般的である。また、記録装置には、その装置固有の記録レートが定められている。従来の民生用映像蓄積メディアで、デジタル放送由来のデジタルビデオ信号を記録する場合、アナログ記録方式であれば、デジタルビデオ信号をデコード後、帯域制限をして記録する。あるいは、MPEG 1 Video、MPEG 2 Video、DV方式をはじめとするデジタル記録方式であれば、1 度デコードされた後に、その装置固有の記録レート・符号化方式で再エンコードされて記録される。

## 【 0 0 0 4 】

しかしながら、このような記録方法は、供給されたビットストリームを1 度デコードし、その後で帯域制限や再エンコードを行って記録するため、画質の劣化を伴う。画像圧縮されたデジタル信号の記録をする場合、入力されたデジタル信号の伝送レートが記録再生装置の記録レートを超えない場合には、供給されたビットストリームをデコードや再エンコードすることなく、そのまま記録する方法が最も画質の劣化が少ない。ただし、画像圧縮されたデジタル信号の伝送レートが記録媒体としてのディスクの記録レートを超える場合には、記録再生装置でデコード後、伝送レートがディスクの記録レートの上限以下になるように、再エンコードをして記録する必要がある。

## 【 0 0 0 5 】

また、入力デジタル信号のビットレートが時間により増減する可変レート方式によって伝送されている場合には、回転ヘッドが固定回転数であるために記録レートが固定レートになるテープ記録方式に比べ、1度バッファにデータを蓄積し、バースト的に記録ができるディスク記録装置が記録媒体の容量をより無駄なく利用できる。

## 【 0 0 0 6 】

以上のように、デジタル放送が主流となる将来においては、データストリーマのように放送信号をデジタル信号のまま、デコードや再エンコードすることなく記録し、記録媒体としてディスクを使用した記録再生装置が求められると予測される。

## 【 0 0 0 7 】

## 【発明が解決しようとする課題】

上述したような記録装置において記録媒体に記録されたデータを再生する際、所定のピクチャまで再生し、そのピクチャから時間的に離れた位置に位置するピクチャを続けて再生するといった、いわゆるスキップ再生というのがある。スキップ再生を行った際、再生する映像に時間的な連続性が途切れてしまことがあるといった課題があった。

## 【 0 0 0 8 】

本発明はこのような状況に鑑みてなされたものであり、再生区間おける動画像の連続性を保つように再生できるようにすることを目的とする。

## 【 0 0 0 9 】

## 【課題を解決するための手段】

請求項1に記載の情報処理装置は、第1のAVストリームから第2のAVストリームへ連続的に再生されるように指示された場合、第1のAVストリームの所定の部分と第2のAVストリームの所定の部分から構成され、第1のAVストリームから第2のAVストリームに再生が切り換えられるとき再生される第3のAVストリームを生成する第1の生成手段と、第1の生成手段により生成された第3のAVストリームの情報として、第1のAVストリームから第3のAVストリームに再生が切り替わるとき再生される第1のAVストリームのソースパケットのアドレスの情報と、第



3のAVストリームから第2のAVストリームに再生が切り替わるとき再生される第2のAVストリームのソースパケットのアドレスの情報から構成されるアドレス情報を生成する第2の生成手段と、第1の生成手段により生成された第3のAVストリームと、第2の生成手段により生成されたアドレス情報を記録する記録手段とを含むことを特徴とする。

## 【 0 0 1 0 】

前記第2の生成手段により生成されたアドレス情報に含まれる第1のAVストリームのソースパケットのアライバルタイムスタンプと、第3のAVストリームの最初に位置するソースパケットのアライバルタイムスタンプは連続しており、かつ、第2の生成手段により生成されたアドレス情報に含まれる第2のAVストリームのソースパケットのアライバルタイムスタンプと、第3のAVストリームの最後に位置するソースパケットのアライバルタイムスタンプは連続しているようにすることができる。

## 【 0 0 1 1 】

前記第3のAVストリーム内のソースパケットのアライバルタイムスタンプには、ただ1つの不連続点が存在するようにすることができる。

## 【 0 0 1 2 】

請求項4に記載の情報処理方法は、第1のAVストリームから第2のAVストリームへ連続的に再生されるように指示された場合、第1のAVストリームの所定の部分と第2のAVストリームの所定の部分から構成され、第1のAVストリームから第2のAVストリームに再生が切り換えられるとき再生される第3のAVストリームを生成する第1の生成ステップと、第1の生成ステップの処理で生成された第3のAVストリームの情報として、第1のAVストリームから第3のAVストリームに再生が切り替わるとき再生される第1のAVストリームのソースパケットのアドレスの情報と、第3のAVストリームから第2のAVストリームに再生が切り替わるとき再生される第2のAVストリームのソースパケットのアドレスの情報から構成されるアドレス情報を生成する第2の生成ステップとを含むことを特徴とする。

## 【 0 0 1 3 】

請求項5に記載の記録媒体のプログラムは、第1のAVストリームから第のAVス

トリームへ連続的に再生されるように指示された場合、第 1 の AV ストリームの所定の部分と第 2 の AV ストリームの所定の部分から構成され、第 1 の AV ストリームから第 2 の AV ストリームに再生が切り換えられるとき再生される第 3 の AV ストリームを生成する第 1 の生成ステップと、第 1 の生成ステップの処理で生成された第 3 の AV ストリームの情報として、第 1 の AV ストリームから第 3 の AV ストリームに再生が切り替わるとき再生される第 1 の AV ストリームのソースパケットのアドレスの情報と、第 3 の AV ストリームから第 2 の AV ストリームに再生が切り替わるとき再生される第 2 の AV ストリームのソースパケットのアドレスの情報から構成されるアドレス情報を生成する第 2 の生成ステップとを含むことを特徴とする。

## 【 0 0 1 4 】

請求項 1 に記載の情報処理装置、請求項 4 に記載の情報処理方法、および請求項 5 に記載の記録媒体においては、第 1 の AV ストリームから第 2 の AV ストリームへ連続的に再生されるように指示された場合、第 1 の AV ストリームの所定の部分と第 2 の AV ストリームの所定の部分から構成され、第 1 の AV ストリームから第 2 の AV ストリームに再生が切り換えられるとき再生される第 3 の AV ストリームが生成され、第 1 の生成手段により生成された第 3 の AV ストリームの情報として、第 1 の AV ストリームから第 3 の AV ストリームに再生が切り替わるとき再生される第 1 の AV ストリームのソースパケットのアドレスの情報と、第 3 の AV ストリームから第 2 の AV ストリームに再生が切り替わるとき再生される第 2 の AV ストリームのソースパケットのアドレスの情報から構成されるアドレス情報が生成される。

## 【 0 0 1 5 】

## 【発明の実施の形態】

以下に、本発明の実施の形態について、図面を参照して説明する。図 1 は、本発明を適用した記録再生装置 1 の内部構成例を示す図である。まず、外部から入力された信号を記録媒体に記録する動作を行う部分の構成について説明する。記録再生装置 1 は、アナログデータ、または、デジタルデータを入力し、記録することができる構成とされている。

## 【 0 0 1 6 】

端子 1 1 には、アナログのビデオ信号が、端子 1 2 には、アナログのオーディ

オ信号が、それぞれ入力される。端子 1 1 に入力されたビデオ信号は、解析部 1 4 と AV エンコーダ 1 5 に、それぞれ出力される。端子 1 2 に入力されたオーディオ信号は、AV エンコーダ 1 5 に出力される。解析部 1 4 は、入力されたビデオ信号からシーンチェンジなどの特徴点を抽出する。

## 【 0 0 1 7 】

AV エンコーダ 1 5 は、入力されたビデオ信号とオーディオ信号を、それぞれ符号化し、符号化ビデオストリーム(V)、符号化オーディオストリーム(A)、および AV 同期等のシステム情報(S)をマルチプレクサ 1 6 に出力する。

## 【 0 0 1 8 】

符号化ビデオストリームは、例えば、MPEG (Moving Picture Expert Group) 2 方式により符号化されたビデオストリームであり、符号化オーディオストリームは、例えば、MPEG 1 方式により符号化されたオーディオストリームや、ドルビー AC3 方式により符号化されたオーディオストリーム等である。マルチプレクサ 1 6 は、入力されたビデオおよびオーディオのストリームを、入力システム情報に基づいて多重化して、スイッチ 1 7 を介して多重化ストリーム解析部 1 8 とソースパケッタイザ 1 9 に出力する。

## 【 0 0 1 9 】

多重化ストリームは、例えば、MPEG2 トランスポートストリームや MPEG2 プログラムストリームである。ソースパケッタイザ 1 9 は、入力された多重化ストリームを、そのストリームを記録させる記録媒体 1 0 0 のアプリケーションフォーマットに従って、ソースパケットから構成される AV ストリームを符号化する。AV ストリームは、ECC (誤り訂正) 符号化部 2 0、変調部 2 1 で所定の処理が施され、書き込み部 2 2 に出力される。書き込み部 2 2 は、制御部 2 3 から出力される制御信号に基づいて、記録媒体 1 0 0 に AV ストリームファイルを書き込む (記録する)。

## 【 0 0 2 0 】

デジタルインタフェースまたはデジタルテレビジョンチューナから入力されるデジタルテレビジョン放送等のトランスポートストリームは、端子 1 3 に入力される。端子 1 3 に入力されたトランスポートストリームの記録方式には、2 通り

あり、それらは、トランスペアレントに記録する方式と、記録ビットレートを下げるなどの目的のために再エンコードをした後に記録する方式である。記録方式の指示情報は、ユーザインターフェースとしての端子 2 4 から制御部 2 3 へ入力される。

#### 【 0 0 2 1 】

入力トランスポートストリームをトランスペアレントに記録する場合、端子 1 3 に入力されたトランスポートストリームは、スイッチ 2 5 を介して多重化ストリーム解析部 1 8 と、ソースパケットタイザ 1 9 に出力される。これ以降の記録媒体 1 0 0 へ AV ストリームが記録されるまでの処理は、上述の入力オーディオ浸透とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

#### 【 0 0 2 2 】

入力トランスポートストリームを再エンコードした後に記録する場合、端子 1 3 に入力されたトランスポートストリームは、スイッチ 2 5 を介してデマルチプレクサ 2 6 に入力される。デマルチプレクサ 2 6 は、入力されたトランスポートストリームに対してデマルチプレクス処理を施し、ビデオストリーム(V)、オーディオストリーム(A)、およびシステム情報(S)を抽出する。

#### 【 0 0 2 3 】

デマルチプレクサ 2 6 により抽出されたストリーム(情報)のうち、ビデオストリームは AV デコーダ 2 7 に、オーディオストリームとシステム情報はマルチプレクサ 1 6 に、それぞれ出力される。AV デコーダ 2 7 は、入力されたビデオストリームを復号し、その再生ビデオ信号を AV エンコーダ 1 5 に出力する。AV エンコーダ 1 5 は、入力ビデオ信号を符号化し、符号化ビデオストリーム(V)をマルチプレクサ 1 6 に出力する。

#### 【 0 0 2 4 】

一方、デマルチプレクサ 2 6 から出力され、マルチプレクサ 1 6 に入力されたオーディオストリームとシステム情報、および、AV エンコーダ 1 5 から出力されたビデオストリームは、入力システム情報に基づいて、多重化されて、多重化ストリームとして多重化ストリーム解析部 1 8 とソースパケットタイザ 1 9 にスイ

ッチ 1 7 を介して出力される。これ以後の記録媒体 1 0 0 へ AV ストリームが記録されるまでの処理は、上述の入力オーディオ信号とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

#### 【 0 0 2 5 】

本実施の形態の記録再生装置 1 は、AV ストリームのファイルを記録媒体 1 0 0 に記録すると共に、そのファイルを説明するアプリケーションデータベース情報も記録する。アプリケーションデータベース情報は、制御部 2 3 により作成される。制御部 2 3 への入力情報は、解析部 1 4 からの動画像の特徴情報、多重化ストリーム解析部 1 8 からの AV ストリームの特徴情報、および端子 2 4 から入力されるユーザからの指示情報である。

#### 【 0 0 2 6 】

解析部 1 4 から供給される動画像の特徴情報は、入力動画像信号の中の特徴的な画像に関する情報であり、例えば、プログラムの開始点、シーンチェンジ点、コマーシャル (CM) の開始・終了点などの指定情報 (マーク) であり、また、その指定場所の画像のサムネイル画像の情報も含まれる。

#### 【 0 0 2 7 】

多重化ストリーム解析部 1 8 からの AV ストリームの特徴情報は、記録される AV ストリームの符号化情報に関する情報であり、例えば、AV ストリーム内の I ピクチャのアドレス情報、AV ストリームの符号化パラメータ、AV ストリームの中の符号化パラメータの変化点情報、ビデオストリームの中の特徴的な画像に関する情報 (マーク) などである。

#### 【 0 0 2 8 】

端子 2 4 からのユーザの指示情報は、AV ストリームの中の、ユーザが指定した再生区間の指定情報、その再生区間の内容を説明するキャラクター文字、ユーザが好みのシーンにセットするブックマークやリジューム点の情報などである。

#### 【 0 0 2 9 】

制御部 2 3 は、上記の入力情報に基づいて、AV ストリームのデータベース (Clip)、AV ストリームの再生区間 (PlayItem) をグループ化したもの (PlayList) のデータベース、記録媒体 1 0 0 の記録内容の管理情報 (info.dvr)、およびサムネ

イル画像の情報を作成する。これらの情報から構成されるアプリケーションデータベース情報は、AVストリームと同様にして、ECC符号化部 2 0、変調部 2 1 で処理されて、書き込み部 2 2 へ入力される。書き込み部 2 2 は、制御部 2 3 から出力される制御信号に基づいて、記録媒体 1 0 0 へデータベースファイルを記録する。

#### 【 0 0 3 0 】

上述したアプリケーションデータベース情報についての詳細は後述する。

#### 【 0 0 3 1 】

このようにして記録媒体 1 0 0 に記録されたAVストリームファイル（画像データと音声データのファイル）と、アプリケーションデータベース情報が再生される場合、まず、制御部 2 3 は、読み出し部 2 8 に対して、記録媒体 1 0 0 からアプリケーションデータベース情報を読み出すように指示する。そして、読み出し部 2 8 は、記録媒体 1 0 0 からアプリケーションデータベース情報を読み出し、そのアプリケーションデータベース情報は、復調部 2 9、ECC復号部 3 0 の処理を経て、制御部 2 3 へ入力される。

#### 【 0 0 3 2 】

制御部 2 3 は、アプリケーションデータベース情報に基づいて、記録媒体 1 0 0 に記録されているPlayListの一覧を端子 2 4 のユーザインターフェースへ出力する。ユーザは、PlayListの一覧から再生したいPlayListを選択し、再生を指定されたPlayListに関する情報が制御部 2 3 へ入力される。制御部 2 3 は、そのPlayListの再生に必要なAVストリームファイルの読み出しを、読み出し部 2 8 に指示する。読み出し部 2 8 は、その指示に従い、記録媒体 1 0 0 から対応するAVストリームを読み出し復調部 2 9 に出力する。復調部 2 9 に入力されたAVストリームは、所定の処理が施されることにより復調され、さらにECC復号部 3 0 の処理を経て、ソースデパケッタイザ 3 1 出力される。

#### 【 0 0 3 3 】

ソースデパケッタイザ 3 1 は、記録媒体 1 0 0 から読み出され、所定の処理が施されたアプリケーションフォーマットのAVストリームを、デマルチプレクサ 2 6 に出力できるストリームに変換する。デマルチプレクサ 2 6 は、制御部 2 3 に

より指定されたAVストリームの再生区間(PlayItem)を構成するビデオストリーム(V)、オーディオストリーム(A)、およびAV同期等のシステム情報(S)を、AVデコーダ27に出力する。AVデコーダ27は、ビデオストリームとオーディオストリームを復号し、再生ビデオ信号と再生オーディオ信号を、それぞれ対応する端子32と端子33から出力する。

#### 【0034】

また、ユーザインタフェースとしての端子24から、ランダムアクセス再生や特殊再生を指示する情報が入力された場合、制御部23は、AVストリームのデータベース(Clip)の内容に基づいて、記憶媒体100からのAVストリームの読み出し位置を決定し、そのAVストリームの読み出しを、読み出し部28に指示する。例えば、ユーザにより選択されたPlayListを、所定の時刻から再生する場合、制御部23は、指定された時刻に最も近いタイムスタンプを持つIピクチャからのデータを読み出すように読み出し部28に指示する。

#### 【0035】

また、ユーザによって高速再生(Fast-forward playback)が指示された場合、制御部23は、AVストリームのデータベース(Clip)に基づいて、AVストリームの中のI-ピクチャデータを順次連続して読み出すように読み出し部28に指示する。

#### 【0036】

読み出し部28は、指定されたランダムアクセスポイントからAVストリームのデータを読み出し、読み出されたデータは、後段の各部の処理を経て再生される。

#### 【0037】

次に、ユーザが、記録媒体100に記録されているAVストリームの編集をする場合を説明する。ユーザが、記録媒体100に記録されているAVストリームの再生区間を指定して新しい再生経路を作成したい場合、例えば、番組Aという歌番組から歌手Aの部分を再生し、その後続けて、番組Bという歌番組の歌手Aの部分を再生したいといった再生経路を作成したい場合、ユーザインタフェースとしての端子24から再生区間の開始点(イン点)と終了点(アウト点)の情報が制

御部 2 3 に入力される。制御部 2 3 は、AV ストリームの再生区間 (PlayItem) をグループ化したもの (PlayList) のデータベースを作成する。

## 【 0 0 3 8 】

ユーザが、記録媒体 1 0 0 に記録されている AV ストリームの一部を消去したい場合、ユーザインタフェースとしての端子 2 4 から消去区間のイン点とアウト点の情報が制御部 2 3 に入力される。制御部 2 3 は、必要な AV ストリーム部分だけを参照するように PlayList のデータベースを変更する。また、AV ストリームの不必要なストリーム部分を消去するように、書き込み部 2 2 に指示する。

## 【 0 0 3 9 】

ユーザが、記録媒体 1 0 0 に記録されている AV ストリームの再生区間を指定して新しい再生経路を作成したい場合であり、かつ、それぞれの再生区間をシームレスに接続したい場合について説明する。このような場合、制御部 2 3 は、AV ストリームの再生区間 (PlayItem) をグループ化したもの (PlayList) のデータベースを作成し、さらに、再生区間の接続点付近のビデオストリームの部分的な再エンコードと再多重化を行う。

## 【 0 0 4 0 】

まず、端子 2 4 から再生区間のイン点のピクチャの情報と、アウト点のピクチャの情報が制御部 2 3 へ入力される。制御部 2 3 は、読み出し部 2 8 にイン点側ピクチャとアウト点側のピクチャを再生するために必要なデータの読み出しを指示する。そして、読み出し部 2 8 は、記録媒体 1 0 0 からデータを読み出し、そのデータは、復調部 2 9、ECC 復号部 3 0、ソースデパケッタイザ 3 1 を経て、デマルチプレクサ 2 6 に出力される。

## 【 0 0 4 1 】

制御部 2 3 は、デマルチプレクサ 2 6 に入力されたデータを解析して、ビデオストリームの再エンコード方法 (picture\_coding\_type の変更、再エンコードする符号化ビット量の割り当て) と、再多重化方式を決定し、その方式を AV エンコーダ 1 5 とマルチプレクサ 1 6 に供給する。

## 【 0 0 4 2 】

次に、デマルチプレクサ 2 6 は、入力されたストリームをビデオストリーム (V



)、オーディオストリーム(A)、およびシステム情報(S)に分離する。ビデオストリームは、「AVデコーダ27に入力されるデータ」と「マルチプレクサ16に入力されるデータ」がある。前者のデータは、再エンコードするために必要なデータであり、これはAVデコーダ27で復号され、復号されたピクチャはAVエンコーダ15で再エンコードされて、ビデオストリームにされる。後者のデータは、再エンコードをしないで、オリジナルのストリームからコピーされるデータである。オーディオストリーム、システム情報については、直接、マルチプレクサ16に入力される。

#### 【0043】

マルチプレクサ16は、制御部23から入力された情報に基づいて、入力ストリームを多重化し、多重化ストリームを出力する。多重化ストリームは、ECC符号化部20、変調部21で処理されて、書き込み部22に入力される。書き込み部22は、制御部23から供給される制御信号に基づいて、記録媒体100にAVストリームを記録する。

#### 【0044】

以下に、アプリケーションデータベース情報や、その情報に基づく再生、編集といった操作に関する説明をする。図2は、アプリケーションフォーマットの構造を説明する図である。アプリケーションフォーマットは、AVストリームの管理のためにPlayListとClipの2つのレイヤをもつ。Volume Informationは、ディスク内のすべてのClipとPlayListの管理をする。ここでは、1つのAVストリームとその付属情報のペアを1つのオブジェクトと考え、それをClipと称する。AVストリームファイルはClip AV stream fileと称し、その付属情報は、Clip Information fileと称する。

#### 【0045】

1つのClip AV stream fileは、MPEG2トランスポートストリームをアプリケーションフォーマットによって規定される構造に配置したデータをストアする。一般的に、ファイルは、バイト列として扱われるが、Clip AV stream fileのコンテンツは、時間軸上に展開され、Clipの中のエン트리ポイントは、主に時間ベースで指定される。所定のClipへのアクセスポイントのタイムスタンプが与えら

れた時、Clip Information fileは、Clip AV stream fileの中でデータの読み出しを開始すべきアドレス情報を見つけるために役立つ。

## 【 0 0 4 6 】

PlayListについて、図 3 を参照して説明する。PlayListは、Clipの中からユーザが見たい再生区間を選択し、それを簡単に編集することができるようにするために設けられている。1つのPlayListは、Clipの中の再生区間の集まりである。所定のClipの中の1つの再生区間は、PlayItemと呼ばれ、それは、時間軸上のイン点 (IN) とアウト点 (OUT) の対で表される。従って、PlayListは、複数のPlayItemが集まることにより構成される。

## 【 0 0 4 7 】

PlayListには、2つのタイプがある。1つは、Real PlayListであり、もう1つは、Virtual PlayListである。Real PlayListは、それが参照しているClipのストリーム部分を共有している。すなわち、Real PlayListは、その参照しているClipのストリーム部分に相当するデータ容量をディスクの中で占め、Real PlayListが消去された場合、それが参照しているClipのストリーム部分もまたデータが消去される。

## 【 0 0 4 8 】

Virtual PlayListは、Clipのデータを共有していない。従って、Virtual PlayListが変更または消去されたとしても、Clipの内容には何も変化が生じない。

## 【 0 0 4 9 】

次に、Real PlayListの編集について説明する。図 4 (A) は、Real PlayListのクリエイト(create: 作成)に関する図であり、AVストリームが新しいClipとして記録される場合、そのClip全体を参照するReal PlayListが新たに作成される操作である。

## 【 0 0 5 0 】

図 4 (B) は、Real PlayListのディバイド(divide: 分割)に関する図であり、Real PlayListが所望な点で分けられて、2つのReal PlayListに分割される操作である。この分割という操作は、例えば、1つのPlayListにより管理される1つのクリップ内に、2つの番組が管理されているような場合に、ユーザが1つ1

つの番組として登録（記録）し直したいといったようなときに行われる。この操作により、Clipの内容が変更される（Clip自体が分割される）ことはない。

#### 【 0 0 5 1 】

図 4（C）は、Real PlayListのコンバイン(combine：結合)に関する図であり、2つのReal PlayListを結合して、1つの新しいReal PlayListにする操作である。この結合という操作は、例えば、ユーザが2つの番組を1つの番組として登録し直したいといったようなときに行われる。この操作により、Clipが変更される（Clip自体が1つにされる）ことはない。

#### 【 0 0 5 2 】

図 5（A）は、Real PlayList全体のデリート(delete：削除)に関する図であり、所定のReal PlayList全体を消去する操作がされた場合、削除されたReal PlayListが参照するClipの、対応するストリーム部分も削除される。

#### 【 0 0 5 3 】

図 5（B）は、Real PlayListの部分的な削除に関する図であり、Real PlayListの所望な部分が削除された場合、対応するPlayItemが、必要なClipのストリーム部分だけを参照するように変更される。そして、Clipの対応するストリーム部分は削除される。

#### 【 0 0 5 4 】

図 5（C）は、Real PlayListのミニマイズ(Minimize：最小化)に関する図であり、Real PlayListに対応するPlayItemを、Virtual PlayListに必要なClipのストリーム部分だけを参照するようにする操作である。Virtual PlayList にとって不必要なClipの、対応するストリーム部分は削除される。

#### 【 0 0 5 5 】

上述したような操作により、Real PlayListが変更されて、そのReal PlayListが参照するClipのストリーム部分が削除された場合、その削除されたClipを使用しているVirtual PlayListが存在し、そのVirtual PlayListにおいて、削除されたClipにより問題が生じる可能性がある。

#### 【 0 0 5 6 】

そのようなことが生じないように、ユーザに、削除という操作に対して、「そ

のReal Playlistが参照しているClipのストリーム部分を参照しているVirtual Playlistが存在し、もし、そのReal Playlistが消去されると、そのVirtual Playlistもまた消去されることになるが、それでも良いか？」といったメッセージなどを表示させることにより、確認（警告）を促した後に、ユーザの指示により削除の処理を実行、または、キャンセルする。または、Virtual Playlistを削除する代わりに、Real Playlistに対してミニマイズの操作が行われるようにする。

## 【 0 0 5 7 】

次にVirtual Playlistに対する操作について説明する。Virtual Playlistに対して操作が行われたとしても、Clipの内容が変更されることはない。図6は、アセンブル(Assemble) 編集 (IN-OUT 編集)に関する図であり、ユーザが見たいと所望した再生区間のPlayItemを作り、Virtual Playlistを作成するといった操作である。PlayItem間のシームレス接続が、アプリケーションフォーマットによりサポートされている（後述）。

## 【 0 0 5 8 】

図6（A）に示したように、2つのReal Playlist 1, 2と、それぞれのReal Playlistに対応するClip 1, 2が存在している場合に、ユーザがReal Playlist 1内の所定の区間（In 1乃至Out 1までの区間：PlayItem 1）を再生区間として指示し、続けて再生する区間として、Real Playlist 2内の所定の区間（In 2乃至Out 2までの区間：PlayItem 2）を再生区間として指示したとき、図6（B）に示すように、PlayItem 1とPlayItem 2から構成される1つのVirtual Playlistが作成される。

## 【 0 0 5 9 】

次に、Virtual Playlist の再編集(Re-editing)について説明する。再編集には、Virtual Playlistの中のイン点やアウト点の変更、Virtual Playlistへの新しいPlayItemの挿入(insert)や追加(append)、Virtual Playlistの中のPlayItemの削除などがある。また、Virtual Playlistそのものを削除することもできる。

## 【 0 0 6 0 】

図7は、Virtual Playlistへのオーディオのアフレコ(Audio dubbing (post recording))に関する図であり、Virtual Playlistへのオーディオのアフレコをサ

ブパスとして登録する操作のことである。このオーディオのアフレコは、アプリケーションフォーマットによりサポートされている。Virtual PlayListのメインパスのAVストリームに、付加的なオーディオストリームが、サブパスとして付加される。

#### 【0061】

Real PlayListとVirtual PlayListで共通の操作として、図8に示すようなPlayListの再生順序の変更(Moving)がある。この操作は、ディスク(ボリューム)の中でのPlayListの再生順序の変更であり、アプリケーションフォーマットにおいて定義されるTable Of PlayList (図20などを参照して後述する)によってサポートされる。この操作により、Clipの内容が変更されるようなことはない。

#### 【0062】

次に、マーク (Mark) について説明する。マークは、ClipおよびPlayListの中のハイライトや特徴的な時間を指定するために設けられている。Clipに付加されるマークは、AVストリームの内容に起因する特徴的なシーンを指定する、例えば、シーンチェンジ点などである。PlayListを再生する時、そのPlayListが参照するClipのマークを参照して、使用する事ができる。

#### 【0063】

PlayListに付加されるマークは、主にユーザによってセットされる、例えば、ブックマークやリジューム点などである。ClipまたはPlayListにマークをセットすることは、マークの時刻を示すタイムスタンプをマークリストに追加することにより行われる。また、マークを削除することは、マークリストの中から、そのマークのタイムスタンプを除去する事である。従って、マークの設定や削除により、AVストリームは何の変更もされない。

#### 【0064】

次にサムネイルについて説明する。サムネイルは、Volume、PlayList、およびClipに付加される静止画である。サムネイルには、2つの種類があり、1つは、内容を表す代表画としてのサムネイルである。これは主としてユーザがカーソル(不図示)などを操作して見たいものを選択するためのメニュー画面で使われるものである。もう1つは、マークが指しているシーンを表す画像である。

## 【0065】

Volumeと各Playlistは代表画を持つことができるようにする必要がある。Volumeの代表画は、ディスク（記録媒体100、以下、記録媒体100はディスク状のものであるとし、適宜、ディスクと記述する）を記録再生装置1の所定の場所にセットした時に、そのディスクの内容を表す静止画を最初に表示する場合などに用いられることを想定している。Playlistの代表画は、Playlistを選択するメニュー画面において、Playlistの内容を表すための静止画として用いられることを想定している。

## 【0066】

Playlistの代表画として、Playlistの最初の画像をサムネイル（代表画）にすることが考えられるが、必ずしも再生時刻0の先頭の画像が内容を表す上で最適な画像とは限らない。そこで、Playlistのサムネイルとして、任意の画像をユーザが設定できるようにする。以上2種類のサムネイルをメニューサムネイルと称する。メニューサムネイルは頻繁に表示されるため、ディスクから高速に読み出される必要がある。このため、すべてのメニューサムネイルを1つのファイルに格納することが効率的である。メニューサムネイルは、必ずしもボリューム内の動画から抜き出したピクチャである必要はなく、図10に示すように、パーソナルコンピュータやデジタルスチルカメラから取り込こまれた画像でもよい。

## 【0067】

一方、ClipとPlaylistには、複数個のマークを打てる必要があり、マーク位置の内容を知るためにマーク点の画像を容易に見ることが出来るようにする必要がある。このようなマーク点を表すピクチャをマークサムネイル（Mark Thumbnail）と称する。従って、サムネイルの元となる画像は、外部から取り込んだ画像よりも、マーク点の画像を抜き出したものが主となる。

## 【0068】

図11は、Playlistに付けられるマークと、そのマークサムネイルの関係について示す図であり、図12は、Clipに付けられるマークと、そのマークサムネイルの関係について示す図である。マークサムネイルは、メニューサムネイルと異なり、Playlistの詳細を表す時に、サブメニュー等で使われるため、短いアクセ

ス時間で読み出されるようなことは要求されない。そのため、サムネイルが必要になる度に、記録再生装置1がファイルを開き、そのファイルの一部を読み出すことで多少時間がかかっても、問題にはならない。

#### 【0069】

また、ボリューム内に存在するファイル数を減らすために、すべてのマークサムネイルは1つのファイルに格納するのがよい。Playlistはメニューサムネイル1つと複数のマークサムネイルを有することができるが、Clipは直接ユーザが選択する必要性がない（通常、Playlist経由で指定する）ため、メニューサムネイルを設ける必要はない。

#### 【0070】

図13は、上述したことを考慮した場合のメニューサムネイル、マークサムネイル、Playlist、およびClipの関係について示した図である。メニューサムネイルファイルには、Playlist毎に設けられたメニューサムネイルがファイルされている。メニューサムネイルファイルには、ディスクに記録されているデータの内容を代表するボリュームサムネイルが含まれている。マークサムネイルファイルは、各Playlist毎と各Clip毎に作成されたサムネイルがファイルされている。

#### 【0071】

次に、CPI (Characteristic Point Information) について説明する。CPIは、Clipインフォメーションファイルに含まれるデータであり、主に、それはClipへのアクセスポイントのタイムスタンプが与えられた時、Clip AV stream fileの中でデータの読み出しを開始すべきデータアドレスを見つけるために用いられる。本実施の形態では、2種類のCPIを用いる。1つは、EP\_mapであり、もう一つは、TU\_mapである。

#### 【0072】

EP\_mapは、エントリーポイント(EP)データのリストであり、それはエレメンタリーストリームおよびトランスポートストリームから抽出されたものである。これは、AVストリームの中でデコードを開始すべきエントリーポイントの場所を見つけるためのアドレス情報を持つ。1つのEPデータは、プレゼンテーションタイムスタンプ(PTS)と、そのPTSに対応するアクセスユニットのAVストリームの中

のデータアドレスの対で構成される。

【0073】

EP\_mapは、主に2つの目的のために使用される。第1に、PlayListの中でプレゼンテーションタイムスタンプによって参照されるアクセスユニットのAVストリームの中のデータアドレスを見つけるために使用される。第2に、ファーストフォワード再生やファーストリバース再生のために使用される。記録再生装置1が、入力AVストリームを記録する場合、そのストリームのシンタクスを解析することができるとき、EP\_mapが作成され、ディスクに記録される。

【0074】

TU\_mapは、デジタルインタフェースを通して入力されるトランスポートパケットの到着時刻に基づいたタイムユニット (TU) データのリストを持つ。これは、到着時刻ベースの時間とAVストリームの中のデータアドレスとの関係を与える。記録再生装置1が、入力AVストリームを記録する場合、そのストリームのシンタクスを解析することができないとき、TU\_mapが作成され、ディスクに記録される。

【0075】

本実施の形態では、セルフエンコードのストリームフォーマット (SESF) を定義する。SESFは、アナログ入力信号を符号化する目的、およびデジタル入力信号 (例えばDV) をデコードしてからMPEG2トランスポートストリームに符号化する場合に用いられる。

【0076】

SESFは、MPEG-2トランスポートストリームおよびAVストリームについてのエレメンタリーストリームの符号化制限を定義する。記録再生装置1が、SESFストリームをエンコードし、記録する場合、EP\_mapが作成され、ディスクに記録される。

【0077】

デジタル放送のストリームは、次に示す方式のうちのいずれかが用いられて記録媒体100に記録される。まず、デジタル放送のストリームをSESFストリームにトランスコーディングする。この場合、記録されたストリームは、SESFに準拠



しなければならない。この場合、EP\_mapが作成されて、ディスクに記録されなければならない。

#### 【0078】

あるいは、デジタル放送ストリームを構成するエレメンタリーストリームを新しいエレメンタリーストリームにトランスコーディングし、そのデジタル放送ストリームの規格化組織が定めるストリームフォーマットに準拠した新しいトランスポートストリームに再多重化する。この場合、EP\_mapが作成されて、ディスクに記録されなければならない。

#### 【0079】

例えば、入力ストリームがISDB（日本のデジタルBS放送の規格名称）準拠のMP EG-2トランスポートストリームであり、それがHDTVビデオストリームとMPEG AACオーディオストリームを含むとする。HDTVビデオストリームをSDTVビデオストリームにトランスコーディングし、そのSDTVビデオストリームとオリジナルのAACオーディオストリームをTSに再多重化する。SDTVストリームと記録されるトランスポートストリームは、共にISDBフォーマットに準拠しなければならない。

#### 【0080】

デジタル放送のストリームが、記録媒体100に記録される際の他の方式として、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、その時にEP\_mapが作成されてディスクに記録される。

#### 【0081】

または、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、その時にTU\_mapが作成されてディスクに記録される。

#### 【0082】

次にディレクトリとファイルについて説明する。以下、記録再生装置1をDVR（Digital Video Recording）と適宜記述する。図14はディスク上のディレクトリ構造の一例を示す図である。DVRのディスク上に必要なディレクトリは、図14に示したように、“DVR”ディレクトリを含むrootディレクトリ、“PLAYLIST”

ディレクトリ、“CLIPINF”ディレクトリ、“M2TS”ディレクトリ、および“DATA”ディレクトリを含む“DVR”ディレクトリである。rootディレクトリの下に、これら以外のディレクトリを作成されるようにしても良いが、それらは、本実施の形態のアプリケーションフォーマットでは、無視されるとする。

## 【 0 0 8 3 】

“DVR”ディレクトリの下には、DVRアプリケーションフォーマットによって規定される全てのファイルとディレクトリがストアされる。“DVR”ディレクトリは、4個のディレクトリを含む。“PLAYLIST”ディレクトリの下には、Real PlayListとVirtual PlayListのデータベースファイルが置かれる。このディレクトリは、PlayListが1つもなくても存在する。

## 【 0 0 8 4 】

“CLIPINF”ディレクトリの下には、Clipのデータベースが置かれる。このディレクトリも、Clipが1つもなくても存在する。“M2TS”ディレクトリの下には、AVストリームファイルが置かれる。このディレクトリは、AVストリームファイルが1つもなくても存在する。“DATA”ディレクトリは、デジタルTV放送などのデータ放送のファイルがストアされる。

## 【 0 0 8 5 】

“DVR”ディレクトリは、次に示すファイルをストアする。“info.dvr”ファイルは、DVRディレクトリの下に作られ、アプリケーションレイヤの全体的な情報をストアする。DVRディレクトリの下には、ただ一つのinfo.dvrがなければならない。ファイル名は、info.dvrに固定されるとする。“menu.thmb”ファイルは、メニューサムネイル画像に関連する情報をストアする。DVRディレクトリの下には、ゼロまたは1つのメニューサムネイルがなければならない。ファイル名は、menu.thmbに固定されるとする。メニューサムネイル画像が1つもない場合、このファイルは、存在しなくても良い。

## 【 0 0 8 6 】

“mark.thmb”ファイルは、マークサムネイル画像に関連する情報をストアする。DVRディレクトリの下には、ゼロまたは1つのマークサムネイルがなければならない。ファイル名は、mark.thmbに固定されるとする。メニューサムネイル画

像が1つもない場合、このファイルは、存在しなくても良い。

#### 【0087】

"PLAYLIST"ディレクトリは、2種類のPlayListファイルをストアするものであり、それらは、Real PlayListとVirtual PlayListである。"xxxxx.rpls" ファイルは、1つのReal PlayListに関連する情報をストアする。それぞれのReal PlayList毎に、1つのファイルが作られる。ファイル名は、"xxxxx.rpls"である。ここで、"xxxxx"は、5個の0乃至9まで数字である。ファイル拡張子は、"rpls"でなければならないとする。

#### 【0088】

"yyyyy.vpls"ファイルは、1つのVirtual PlayListに関連する情報をストアする。それぞれのVirtual PlayList毎に、1つのファイルが作られる。ファイル名は、"yyyyy.vpls"である。ここで、"yyyyy"は、5個の0乃至9まで数字である。ファイル拡張子は、"vpls"でなければならないとする。

#### 【0089】

"CLIPINF"ディレクトリは、それぞれのAVストリームファイルに対応して、1つのファイルをストアする。"zzzzz.clpi" ファイルは、1つのAVストリームファイル(Clip AV stream file または Bridge-Clip AV stream file)に対応するClip Information fileである。ファイル名は、"zzzzz.clpi"であり、"zzzzz"は、5個の0乃至9までの数字である。ファイル拡張子は、"clpi"でなければならないとする。

#### 【0090】

"M2TS"ディレクトリは、AVストリームのファイルをストアする。"zzzzz.m2ts" ファイルは、DVRシステムにより扱われるAVストリームファイルである。これは、Clip AV stream fileまたはBridge-Clip AV streamである。ファイル名は、"zzzzz.m2ts"であり、"zzzzz"は、5個の0乃至9までの数字である。ファイル拡張子は、"m2ts"でなければならないとする。

#### 【0091】

"DATA" ディレクトリは、データ放送から伝送されるデータをストアするものであり、データとは、例えば、XML fileやMHEGファイルなどである。

## 【 0 0 9 2 】

次に、各ディレクトリ（ファイル）のシンタクスとセマンティクスを説明する。まず、“info.dvr”ディレクトリについて説明する。図15は、“info.dvr”ディレクトリのシンタクスを示す図である。“info.dvr”ディレクトリは、3個のオブジェクトから構成され、それらは、DVRVolume()、TableOfPlayLists()、およびMakerPrivateData()である。

## 【 0 0 9 3 】

図15に示したinfo.dvrのシンタクスについて説明するに、TableOfPlayLists\_Start\_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、TableOfPlayList()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

## 【 0 0 9 4 】

MakerPrivateData\_Start\_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。padding\_word（パディングワード）は、info.dvrのシンタクスに従って挿入される。N1とN2は、ゼロまたは任意の正の整数である。それぞれのパディングワードは、任意の値を取るようにしても良い。

## 【 0 0 9 5 】

DVRVolume()は、ボリューム（ディスク）の内容を記述する情報をストアする。図16は、DVRVolume()のシンタクスを示す図である。図16に示したDVRVolume()のシンタクスを説明するに、version\_numberは、このDVRVolume()のバージョンナンバを示す4個のキャラクター文字を示す。version\_numberは、ISO 646に従って、“0045”と符号化される。

## 【 0 0 9 6 】

lengthは、このlengthフィールドの直後からDVRVolume()の最後までのDVRVolume()のバイト数を示す32ビットの符号なし整数で表される。

## 【 0 0 9 7 】

ResumeVolume()は、ボリュームの中で最後に再生したReal PlayListまたはVirtual PlayListのファイル名を記憶している。ただし、Real PlayListまたはVirtual

ual Playlistの再生をユーザが中断した時の再生位置は、PlaylistMark()において定義されるresume-markにストアされる。

## 【 0 0 9 8 】

図 1 7 は、ResumeVolume()のシンタクスを示す図である。図 1 7 に示したResumeVolume()のシンタクスを説明するに、valid\_flagは、この1ビットのフラグが1にセットされている場合、resume\_PlayList\_nameフィールドが有効であることを示し、このフラグが0にセットされている場合、resume\_PlayList\_nameフィールドが無効であることを示す。

## 【 0 0 9 9 】

resume\_PlayList\_nameの10バイトのフィールドは、リジュームされるべきReal PlaylistまたはVirtual Playlistのファイル名を示す。

## 【 0 1 0 0 】

図 1 6 に示したDVRVolume()のシンタクスのなかの、UIAppInfoVolume は、ボリュームについてのユーザインターフェースアプリケーションのパラメータをストアする。図 1 8 は、UIAppInfoVolumeのシンタクスを示す図であり、そのセマンティクスを説明するに、character\_setの8ビットのフィールドは、Volume\_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図 1 9 に示される値に対応する。

## 【 0 1 0 1 】

name\_lengthの8ビットフィールドは、Volume\_nameフィールドの中に示されるボリューム名のバイト長を示す。Volume\_nameのフィールドは、ボリュームの名称を示す。このフィールドの中の左からname\_length数のバイト数が、有効なキャラクター文字であり、それはボリュームの名称を示す。Volume\_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていても良い。

## 【 0 1 0 2 】

Volume\_protect\_flagは、ボリュームの中のコンテンツを、ユーザに制限することなしに見せてよいかどうかを示すフラグである。このフラグが1にセットされている場合、ユーザが正しくPIN番号（パスワード）を入力できたときだけ、

そのボリュームのコンテンツを、ユーザに見せる事（再生される事）が許可される。このフラグが0にセットされている場合、ユーザがPIN番号を入力しなくても、そのボリュームのコンテンツを、ユーザに見せる事が許可される。

## 【 0 1 0 3 】

最初に、ユーザが、ディスクをプレーヤへ挿入した時点において、もしこのフラグが0にセットされているか、または、このフラグが1にセットされていてもユーザがPIN番号を正しく入力できたならば、記録再生装置1は、そのディスクの中のPlayListの一覧を表示させる。それぞれのPlayListの再生制限は、`volume_protect_flag`とは無関係であり、それは`UIAppInfoPlayList()`の中に定義される`playback_control_flag`によって示される。

## 【 0 1 0 4 】

PINは、4個の0乃至9までの数字で構成され、それぞれの数字は、ISO/IEC 646に従って符号化される。`ref_thumbnail_index`のフィールドは、ボリュームに付加されるサムネイル画像の情報を示す。`ref_thumbnail_index`フィールドが、0xFFFFでない値の場合、そのボリュームにはサムネイル画像が付加されており、そのサムネイル画像は、`menu.thum`ファイルの中にストアされている。その画像は、`menu.thum`ファイルの中で`ref_thumbnail_index`の値を用いて参照される。`ref_thumbnail_index`フィールドが、0xFFFFである場合、そのボリュームにはサムネイル画像が付加されていないことを示す。

## 【 0 1 0 5 】

次に図15に示した`info.dvr`のシンタクス内の`TableOfPlayLists()`について説明する。`TableOfPlayLists()`は、`PlayList(Real PlayListとVirtual PlayList)`のファイル名をストアする。ボリュームに記録されているすべてのPlayListファイルは、`TableOfPlayList()`の中に含まれる。`TableOfPlayLists()`は、ボリュームの中のPlayListのデフォルトの再生順序を示す。

## 【 0 1 0 6 】

図20は、`TableOfPlayLists()`のシンタクスを示す図であり、そのシンタクスについて説明するに、`TableOfPlayLists`の`version_number`は、この`TableOfPlayLists`のバージョンナンバーを示す4個のキャラクター文字を示す。`version_num`

erは、ISO 646に従って、“0045”と符号化されなければならない。

#### 【0107】

lengthは、このlengthフィールドの直後からTableOfPlayLists()の最後まで  
TableOfPlayLists()のバイト数を示す32ビットの符号なしの整数である。num  
ber\_of\_PlayListsの16ビットのフィールドは、PlayList\_file\_nameを含むfor-l  
oopのループ回数を示す。この数字は、ボリュームに記録されているPlayListの  
数に等しくなければならない。PlayList\_file\_nameの10バイトの数字は、Play  
Listのファイル名を示す。

#### 【0108】

図21は、TableOfPlayLists()のシンタクスを別実施の構成を示す図である。  
図21に示したシンタクスは、図20に示したシンタクスに、UIAppinfoPlayLis  
t（後述）を含ませた構成とされている。このように、UIAppinfoPlayListを含  
ませた構成とすることで、TableOfPlayListsを読み出すだけで、メニュー画面を作  
成することが可能となる。ここでは、図20に示したシンタクスを用いるとして  
以下の説明をする。

#### 【0109】

図15に示したinfo.dvrのシンタクス内のMakersPrivateDataについて説明す  
る。MakersPrivateDataは、記録再生装置1のメーカーが、各社の特別なアプリケ  
ーションのために、MakersPrivateData()の中にメーカーのプライベートデータを  
挿入できるように設けられている。各メーカーのプライベートデータは、それを定  
義したメーカーを識別するために標準化されたmaker\_IDを持つ。MakersPrivateDat  
a()は、1つ以上のmaker\_IDを含んでも良い。

#### 【0110】

所定のメーカーが、プライベートデータを挿入したい時に、すでに他のメーカーの  
プライベートデータがMakersPrivateData()に含まれていた場合、他のメーカーは  
、既にある古いプライベートデータを消去するのではなく、新しいプライベート  
データをMakersPrivateData()の中に追加するようにする。このように、本実施  
の形態においては、複数のメーカーのプライベートデータが、1つのMakersPrivat  
eData()に含まれることが可能であるようにする。

## 【0 1 1 1】

図 2 2 は、MakersPrivateDataのシンタクスを示す図である。図 2 2 に示したMakersPrivateDataのシンタクスについて説明するに、version\_numberは、このMakersPrivateData()のバージョンナンバを示す4個のキャラクター文字を示す。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からMakersPrivateData()の最後までMakersPrivateData()のバイト数を示す32ビットの符号なし整数を示す。

## 【0 1 1 2】

mpd\_blocks\_start\_addressは、MakersPrivateData()の先頭のバイトからの相対バイト数を単位として、最初のmpd\_block()の先頭バイトアドレスを示す。相対バイト数はゼロからカウントされる。number\_of\_maker\_entriesは、MakersPrivateData()の中に含まれているメーカープライベートデータのエントリー数を与える16ビットの符号なし整数である。MakersPrivateData()の中に、同じmaker\_IDの値を持つメーカープライベートデータが2個以上存在してはならない。

## 【0 1 1 3】

mpd\_block\_sizeは、1024バイトを単位として、1つのmpd\_blockの大きさを与える16ビットの符号なし整数である。例えば、mpd\_block\_size=1ならば、それは1つのmpd\_blockの大きさが1024バイトであることを示す。number\_of\_mpd\_blocksは、MakersPrivateData()の中に含まれるmpd\_blockの数を与える16ビットの符号なし整数である。maker\_IDは、そのメーカープライベートデータを作成したDVRシステムの製造メーカーを示す16ビットの符号なし整数である。maker\_IDに符号化される値は、このDVRフォーマットのライセンスによって指定される。

## 【0 1 1 4】

maker\_model\_codeは、そのメーカープライベートデータを作成したDVRシステムのモデルナンバーコードを示す16ビットの符号なし整数である。maker\_model\_codeに符号化される値は、このフォーマットのライセンスを受けた製造メーカーによって設定される。start\_mpd\_block\_numberは、そのメーカープライベートデータが開始されるmpd\_blockの番号を示す16ビットの符号なし整数である。メーカー



プライベートデータの先頭データは、mpd\_blockの先頭にアラインされなければならない。start\_mpd\_block\_numberは、mpd\_blockのfor-loopの中の変数jに対応する。

## 【0115】

mpd\_lengthは、バイト単位でメーカプライベートデータの大きさを示す32ビットの符号なし整数である。mpd\_blockは、メーカプライベートデータがストアされる領域である。MakersPrivateData()の中のすべてのmpd\_blockは、同じサイズでなければならない。

## 【0116】

次に、Real PlayList fileとVirtual PlayList fileについて、換言すれば、xxxx.rplsとyyyy.vplsについて説明する。図23は、xxxx.rpls (Real PlayList)、または、yyyy.vpls (Virtual PlayList) のシンタクスを示す図である。xxxx.rplsとyyyy.vplsは、同一のシンタクス構成をもつ。xxxx.rplsとyyyy.vplsは、それぞれ、3個のオブジェクトから構成され、それらは、PlayList()、PlayListMark()、およびMakerPrivateData()である。

## 【0117】

PlayListMark\_Start\_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、PlayListMark()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

## 【0118】

MakerPrivateData\_Start\_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

## 【0119】

padding\_word (パディングワード) は、PlayListファイルのシンタクスにしたがって挿入され、N1とN2は、ゼロまたは任意の正の整数である。それぞれのパディングワードは、任意の値を取るようにしても良い。

## 【0120】

ここで、既に、簡便に説明したが、PlayListについてさらに説明する。ディス

ク内にあるすべてのReal Playlistによって、Bridge-Clip（後述）を除くすべてのClipの中の再生区間が参照されていなければならない。かつ、2つ以上のReal Playlistが、それらのPlayItemで示される再生区間を同一のClipの中でオーバーラップさせてはならない。

#### 【0 1 2 1】

図24を参照してさらに説明するに、図24（A）に示したように、全てのClipは、対応するReal Playlistが存在する。この規則は、図24（B）に示したように、編集作業が行われた後においても守られる。従って、全てのClipは、どれかしらのReal Playlistを参照することにより、必ず視聴することが可能である。

#### 【0 1 2 2】

図24（C）に示したように、Virtual Playlistの再生区間は、Real Playlistの再生区間またはBridge-Clipの再生区間の中に含まれていなければならない。どのVirtual Playlistにも参照されないBridge-Clipがディスクの中に存在してはならない。

#### 【0 1 2 3】

Real Playlistは、PlayItemのリストを含むが、SubPlayItemを含んではならない。Virtual Playlistは、PlayItemのリストを含み、Playlist()の中に示されるCPI\_typeがEP\_map typeであり、かつPlaylist\_typeが0（ビデオとオーディオを含むPlaylist）である場合、Virtual Playlistは、ひとつのSubPlayItemを含む事ができる。本実施の形態におけるPlaylist()では、SubPlayItemはオーディオのアフレコの目的にだけに使用される、そして、1つのVirtual Playlistが持つSubPlayItemの数は、0または1でなければならない。

#### 【0 1 2 4】

次に、Playlistについて説明する。図25は、Playlistのシンタクスを示す図である。図25に示したPlaylistのシンタクスを説明するに、version\_numberは、このPlaylist()のバージョンナンバーを示す4個のキャラクター文字である。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からPlaylist()の最後までのPlaylist()

のバイト数を示す 3 2 ビットの符号なし整数である。PlayList\_typeは、このPlayListのタイプを示す 8 ビットのフィールドであり、その一例を図 2 6 に示す。

#### 【 0 1 2 5 】

CPI\_typeは、1 ビットのフラグであり、PlayItem()およびSubPlayItem()によって参照されるClipのCPI\_typeの値を示す。1 つのPlayListによって参照される全てのClipは、それらのCPI()の中に定義されるCPI\_typeの値が同じでなければならない。number\_of\_PlayItemsは、PlayListの中にあるPlayItemの数を示す 1 6 ビットのフィールドである。

#### 【 0 1 2 6 】

所定のPlayItem()に対応するPlayItem\_idは、PlayItem()を含むfor-loopの中で、そのPlayItem()の現れる順番により定義される。PlayItem\_idは、0 から開始される。number\_of\_SubPlayItemsは、PlayListの中にあるSubPlayItemの数を示す 1 6 ビットのフィールドである。この値は、0 または 1 である。付加的なオーディオストリームのパス(オーディオストリームパス)は、サブパスの一種である。

#### 【 0 1 2 7 】

次に、図 2 5 に示したPlayListのシンタクスのUIAppInfoPlayListについて説明する。UIAppInfoPlayListは、PlayListについてのユーザインターフェースアプリケーションのパラメータをストアする。図 2 7 は、UIAppInfoPlayListのシンタクスを示す図である。図 2 7 に示したUIAppInfoPlayListのシンタクスを説明するに、character\_setは、8 ビットのフィールドであり、PlayList\_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図 1 9 に示したテーブルに準拠する値に対応する。

#### 【 0 1 2 8 】

name\_lengthは、8 ビットフィールドであり、PlayList\_nameフィールドの中に示されるPlayList名のバイト長を示す。PlayList\_nameのフィールドは、PlayListの名称を示す。このフィールドの中の左からname\_length数のバイト数が、有効なキャラクター文字であり、それはPlayListの名称を示す。PlayList\_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入ってい

ても良い。

#### 【 0 1 2 9 】

record\_time\_and\_dateは、PlayListが記録された時の日時をストアする56ビットのフィールドである。このフィールドは、年／月／日／時／分／秒について、14個の数字を4ビットのBinary Coded Decimal(BCD)で符号化したものである。例えば、2001/12/23:01:02:03 は、"0x20011223010203"と符号化される。

#### 【 0 1 3 0 】

durationは、PlayListの総再生時間を時間／分／秒の単位で示した24ビットのフィールドである。このフィールドは、6個の数字を4ビットのBinary Coded Decimal(BCD)で符号化したものである。例えば、01:45:30は、"0x014530"と符号化される。

#### 【 0 1 3 1 】

valid\_periodは、PlayListが有効である期間を示す32ビットのフィールドである。このフィールドは、8個の数字を4ビットのBinary Coded Decimal(BCD)で符号化したものである。例えば、記録再生装置1は、この有効期間の過ぎたPlayListを自動消去する、といったように用いられる。例えば、2001/05/07 は、"0x20010507"と符号化される。

#### 【 0 1 3 2 】

maker\_idは、そのPlayListを最後に更新したDVRプレーヤ（記録再生装置1）の製造者を示す16ビットの符号なし整数である。maker\_idに符号化される値は、DVRフォーマットのライセンスによって割り当てられる。maker\_codeは、そのPlayListを最後に更新したDVRプレーヤのモデル番号を示す16ビットの符号なし整数である。maker\_codeに符号化される値は、DVRフォーマットのライセンスを受けた製造者によって決められる。

#### 【 0 1 3 3 】

playback\_control\_flagのフラグが1にセットされている場合、ユーザが正しくPIN番号を入力できた場合にだけ、そのPlayListは再生される。このフラグが0にセットされている場合、ユーザがPIN番号を入力しなくても、ユーザは、そのPlayListを視聴することができる。

## 【0134】

write\_protect\_flagは、図28（A）にテーブルを示すように、1にセットされている場合、write\_protect\_flagを除いて、そのPlayListの内容は、消去および変更されない。このフラグが0にセットされている場合、ユーザは、そのPlayListを自由に消去および変更できる。このフラグが1にセットされている場合、ユーザが、そのPlayListを消去、編集、または上書きする前に、記録再生装置1はユーザに再確認するようなメッセージを表示させる。

## 【0135】

write\_protect\_flagが0にセットされているReal PlayListが存在し、かつ、そのReal PlayListのClipを参照するVirtual PlayListが存在し、そのVirtual PlayListのwrite\_protect\_flagが1にセットされていても良い。ユーザが、Real PlayListを消去しようとする場合、記録再生装置1は、そのReal PlayListを消去する前に、上記Virtual PlayListの存在をユーザに警告するか、または、そのReal PlayListを”Minimize”する。

## 【0136】

is\_played\_flagは、図28（B）に示すように、フラグが1にセットされている場合、そのPlayListは、記録されてから一度は再生されたことを示し、0にセットされている場合、そのPlayListは、記録されてから一度も再生されたことがないことを示す。

## 【0137】

archiveは、図28（C）に示すように、そのPlayListがオリジナルであるか、コピーされたものであるかを示す2ビットのフィールドである。ref\_thumbnail\_indexのフィールドは、PlayListを代表するサムネイル画像の情報を示す。ref\_thumbnail\_indexフィールドが、0xFFFFでない値の場合、そのPlayListには、PlayListを代表するサムネイル画像が付加されており、そのサムネイル画像は、menu.thum ファイルの中にストアされている。その画像は、menu.thumファイルの中でref\_thumbnail\_indexの値を用いて参照される。ref\_thumbnail\_indexフィールドが、0xFFFF である場合、そのPlayListには、PlayListを代表するサムネイル画像が付加されていない。

## 【 0 1 3 8 】

次にPlayItemについて説明する。1つのPlayItem()は、基本的に次のデータを含む。Clipのファイル名を指定するためのClip\_information\_file\_name、Clipの再生区間を特定するためのIN\_timeとOUT\_timeのペア、PlayList()において定義されるCPI\_typeがEP\_map typeである場合、IN\_timeとOUT\_timeが参照するところのSTC\_sequence\_id、および、先行するPlayItemと現在のPlayItemとの接続の状態を示すところのconnection\_conditionである。

## 【 0 1 3 9 】

PlayListが2つ以上のPlayItemから構成される時、それらのPlayItemはPlayListのグローバル時間軸上に、時間のギャップまたはオーバーラップなしに一列に並べられる。PlayList()において定義されるCPI\_typeがEP\_map typeであり、かつ現在のPlayItemがBridgeSequence()を持たない時、そのPlayItemにおいて定義されるIN\_timeとOUT\_timeのペアは、STC\_sequence\_idによって指定される同じSTC連続区間上の時間を指していなければならない。そのような例を図29に示す。

## 【 0 1 4 0 】

図30は、PlayList()において定義されるCPI\_typeがEP\_map typeであり、かつ現在のPlayItemがBridgeSequence()を持つ時、次に説明する規則が適用される場合を示している。現在のPlayItemに先行するPlayItemのIN\_time (図の中でIN\_time1と示されているものは、先行するPlayItemのSTC\_sequence\_idによって指定されるSTC連続区間上の時間を指している。先行するPlayItemのOUT\_time (図の中でOUT\_time1と示されているものは、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このOUT\_timeは、後述する符号化制限に従っていなければならない。

## 【 0 1 4 1 】

現在のPlayItemのIN\_time (図の中でIN\_time2と示されているものは、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このIN\_timeも、後述する符号化制限に従っていなければならない。現在のPlayItemのOUT\_time (図の中でOUT\_time2と示されているも

の)は、現在のPlayItemのSTC\_sequence\_idによって指定されるSTC連続区間上の時間を指している。

#### 【 0 1 4 2 】

図 3 1 に示すように、PlayList()のCPI\_typeがTU\_map typeである場合、PlayItemのIN\_timeとOUT\_timeのペアは、同じClip AVストリーム上の時間を指している。

#### 【 0 1 4 3 】

PlayItemのシンタクスは、図 3 2 に示すようになる。図 3 2 に示したPlayItemのシンタクスを説明するに、Clip\_Information\_file\_nameのフィールドは、Clip Information fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip\_stream\_typeは、Clip AV streamを示していなければならない。

#### 【 0 1 4 4 】

STC\_sequence\_idは、8ビットのフィールドであり、PlayItemが参照するSTC連続区間のSTC\_sequence\_idを示す。PlayList()の中で指定されるCPI\_typeがTU\_map typeである場合、この8ビットフィールドは何も意味を持たず、0にセットされる。IN\_timeは、32ビットフィールドであり、PlayItemの再生開始時刻をストアする。IN\_timeのセマンティクスは、図 3 3 に示すように、PlayList()において定義されるCPI\_typeによって異なる。

#### 【 0 1 4 5 】

OUT\_timeは、32ビットフィールドであり、PlayItemの再生終了時刻をストアする。OUT\_timeのセマンティクスは、図 3 4 に示すように、PlayList()において定義されるCPI\_typeによって異なる。

#### 【 0 1 4 6 】

Connection\_Conditionは、図 3 5 に示したような先行するPlayItemと、現在のPlayItemとの間の接続状態を示す2ビットのフィールドである。図 3 6 は、図 3 5 に示したConnection\_Conditionの各状態について説明する図である。

#### 【 0 1 4 7 】

次に、BridgeSequenceInfoについて、図 3 7 を参照して説明する。BridgeSequ

enceInfo()は、現在のPlayItemの付属情報であり、次に示す情報を持つ。Bridge-Clip AV streamファイルとそれに対応するClip Information fileを指定するBridge\_Clip\_Information\_file\_nameを含む。

#### 【0148】

また、先行するPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。このアドレスは、RSPN\_exit\_from\_previous\_Clipと称される。さらに現在のPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。このアドレスは、RSPN\_enter\_to\_current\_Clipと称される。

#### 【0149】

図37において、RSPN\_arrival\_time\_discontinuityは、the Bridge-Clip AV streamファイルの中でアライバルタイムベースの不連続点があるところのソースパケットのアドレスを示す。このアドレスは、ClipInfo()の中において定義される。

#### 【0150】

図38は、BridgeSequenceinfoのシンタクスを示す図である。図38に示したBridgeSequenceinfoのシンタクスを説明するに、Bridge\_Clip\_Information\_file\_nameのフィールドは、Bridge-Clip AV streamファイルに対応するClip Information fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip\_stream\_typeは、'Bridge-Clip AV stream'を示していなければならない。

#### 【0151】

RSPN\_exit\_from\_previous\_Clipの32ビットフィールドは、先行するPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。RSPN\_exit\_from\_previous\_Clipは、ソースパケット番号を単位とする大きさであり、先行するPlayItemが参照するClip AV streamファイルの最初の



ソースパケットからClipInfo()において定義されるoffset\_SPNの値を初期値としてカウントされる。

#### 【 0 1 5 2 】

RSPN\_enter\_to\_current\_Clipの32ビットフィールドは、現在のPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。RSPN\_exit\_from\_previous\_Clipは、ソースパケット番号を単位とする大きさであり、現在のPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset\_SPNの値を初期値としてカウントされる。

#### 【 0 1 5 3 】

次に、SubPlayItemについて、図39を参照して説明する。SubPlayItem()の使用は、PlayList()のCPI\_typeがEP\_map typeである場合だけに許される。本実施の形態においては、SubPlayItemはオーディオのアフレコの目的のためだけに使用されとする。SubPlayItem()は、次に示すデータを含む。まず、PlayListの中のsub pathが参照するClipを指定するためのClip\_information\_file\_nameを含む。

#### 【 0 1 5 4 】

また、Clipの中のsub pathの再生区間を指定するためのSubPath\_IN\_time と SubPath\_OUT\_timeを含む。さらに、main pathの時間軸上でsub pathが再生開始する時刻を指定するためのsync\_PlayItem\_id と sync\_start\_PTS\_of\_PlayItemを含む。sub pathに参照されるオーディオのClip AV streamは、STC不連続点（システムタイムベースの不連続点）を含んではならない。sub pathに使われるClipのオーディオサンプルのクロックは、main pathのオーディオサンプルのクロックにロックされている。

#### 【 0 1 5 5 】

図40は、SubPlayItemのシンタクスを示す図である。図40に示したSubPlayItemのシンタクスを説明するに、Clip\_Information\_file\_nameのフィールドは、Clip Information fileのファイル名を示し、それはPlayListの中でsub pathに

よって使用される。このClip Information fileのClipInfo()において定義されるClip\_stream\_typeは、Clip AV streamを示していなければならない。

#### 【 0 1 5 6 】

SubPath\_typeの8ビットのフィールドは、sub pathのタイプを示す。ここでは、図4 1に示すように、'0x00'しか設定されておらず、他の値は、将来のために確保されている。

#### 【 0 1 5 7 】

sync\_PlayItem\_idの8ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻が含まれるPlayItemのPlayItem\_idを示す。所定のPlayItemに対応するPlayItem\_idの値は、PlayList()において定義される（図2 5参照）。

#### 【 0 1 5 8 】

sync\_start\_PTS\_of\_PlayItemの3 2ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻を示し、sync\_PlayItem\_idで参照されるPlayItem上のPTS(Presentation Time Stamp)の上位3 2ビットを示す。SubPath\_IN\_timeの3 2ビットフィールドは、Sub pathの再生開始時刻をストアする。SubPath\_IN\_timeは、Sub Pathの中で最初のプレゼンテーションユニットに対応する3 3ビット長のPTSの上位3 2ビットを示す。

#### 【 0 1 5 9 】

SubPath\_OUT\_timeの3 2ビットフィールドは、Sub pathの再生終了時刻をストアする。SubPath\_OUT\_timeは、次式によって算出されるPresentation\_end\_TSの値の上位3 2ビットを示す。

$$\text{Presentation\_end\_TS} = \text{PTS\_out} + \text{AU\_duration}$$

ここで、PTS\_outは、SubPathの最後のプレゼンテーションユニットに対応する33ビット長のPTSである。AU\_durationは、SubPathの最後のプレゼンテーションユニットの9 0 kHz単位の表示期間である。

#### 【 0 1 6 0 】

次に、図2 3に示したxxxxx.rplsとyyyyy.vplsのシンタクス内のPlayListMark()について説明する。PlayListについてのマーク情報は、このPlayListMarkにス

トアされる。図 4 2 は、PlayListMarkのシンタクスを示す図である。図 4 2 に示したPlayListMarkのシンタクスについて説明するに、version\_numberは、このPlayListMark()のバージョンナンバを示す 4 個のキャラクター文字である。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。

## 【 0 1 6 1 】

lengthは、このlengthフィールドの直後からPlayListMark()の最後までPlayListMark()のバイト数を示す 3 2 ビットの符号なし整数である。number\_of\_PlayList\_marksは、PlayListMarkの中にストアされているマークの個数を示す 1 6 ビットの符号なし整数である。number\_of\_PlayList\_marks は、0 であってもよい。mark\_typeは、マークのタイプを示す 8 ビットのフィールドであり、図 4 3 に示すテーブルに従って符号化される。

## 【 0 1 6 2 】

mark\_time\_stampの32ビットフィールドは、マークが指定されたポイントを示すタイムスタンプをストアする。mark\_time\_stampのセマンティクスは、図 4 4 に示すように、PlayList()において定義されるCPI\_typeによって異なる。PlayItem\_idは、マークが置かれているところのPlayItemを指定する 8 ビットのフィールドである。所定のPlayItemに対応するPlayItem\_idの値は、PlayList()において定義される（図 2 5 参照）。

## 【 0 1 6 3 】

character\_setの 8 ビットのフィールドは、mark\_nameフィールドに符号化されているキャラクター文字の符号化方法を示す。その符号化方法は、図 1 9 に示した値に対応する。name\_lengthの 8 ビットフィールドは、Mark\_nameフィールドの中に示されるマーク名のバイト長を示す。mark\_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname\_length数のバイト数が、有効なキャラクター文字であり、それはマークの名称を示す。Mark\_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どのような値が設定されても良い。

## 【 0 1 6 4 】

ref\_thumbnail\_indexのフィールドは、マークに付加されるサムネイル画像の

情報を示す。ref\_thumbnail\_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref\_thumbnail\_indexの値を用いて参照される（後述）。ref\_thumbnail\_indexフィールドが、0xFFFFである場合、そのマークにはサムネイル画像が付加されていない事を示す。

## 【 0 1 6 5 】

次に、Clip information fileについて説明する。zzzzz.clpi (Clip information fileファイル) は、図 4 5 に示すように 6 個のオブジェクトから構成される。それらは、ClipInfo()、STC\_Info()、ProgramInfo()、CPI()、ClipMark()、およびMakerPrivateData()である。AVストリーム(Clip AVストリームまたはBridge-Clip AV stream)とそれに対応するClip Informationファイルは、同じ数字列の"zzzzz"が使用される。

## 【 0 1 6 6 】

図 4 5 に示したzzzzz.clpi (Clip information fileファイル) のシンタクスについて説明するに、ClipInfo\_Start\_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipInfo()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

## 【 0 1 6 7 】

STC\_Info\_Start\_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、STC\_Info()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。ProgramInfo\_Start\_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ProgramInfo()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。CPI\_Start\_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、CPI()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。

## 【 0 1 6 8 】

ClipMark\_Start\_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipMark()の先頭アドレスを示す。相対バイト数はゼロか

らカウントされる。MakerPrivateData\_Start\_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、MakerPrivateData ()の先頭アドレスを示す。相対バイト数はゼロからカウントされる。padding\_word (パディングワード) は、zzzzz.clpiファイルのシンタクスにしたがって挿入される。N 1, N 2, N 3, N 4、およびN 5は、ゼロまたは任意の正の整数でなければならない。それぞれのパディングワードは、任意の値がとられるようにしても良い。

#### 【0 1 6 9】

次に、ClipInfoについて説明する。図 4 6 は、ClipInfoのシンタクスを示す図である。ClipInfo()は、それに対応するAVストリームファイル (Clip AVストリームまたはBridge-Clip AVストリームファイル) の属性情報をストアする。

#### 【0 1 7 0】

図 4 6 に示したClipInfoのシンタクスについて説明するに、version\_numberは、このClipInfo()のバージョンナンバーを示す4個のキャラクター文字である。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からClipInfo()の最後までClipInfo()のバイト数を示す32ビットの符号なし整数である。Clip\_stream\_typeの8ビットのフィールドは、図 4 7 に示すように、Clip Informationファイルに対応するAVストリームのタイプを示す。それぞれのタイプのAVストリームのストリームタイプについては後述する。

#### 【0 1 7 1】

offset\_SPNの32ビットのフィールドは、AVストリーム (Clip AVストリームまたはBridge-Clip AVストリーム) ファイルの最初のソースパケットについてのソースパケット番号のオフセット値を与える。AVストリームファイルが最初にディスクに記録される時、このoffset\_SPNは0でなければならない。

#### 【0 1 7 2】

図 4 8 に示すように、AVストリームファイルのはじめの部分が編集によって消去された時、offset\_SPNは、ゼロ以外の値をとっても良い。本実施の形態では、offset\_SPNを参照する相対ソースパケット番号 (相対アドレス) が、しばしば、

RSPN\_xxx (xxxは変形する。例、RSPN\_EP\_start) の形式でシンタクスの中に記述されている。相対ソースパケット番号は、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからoffset\_SPNの値を初期値としてカウントされる。

## 【 0 1 7 3 】

AVストリームファイルの最初のソースパケットから相対ソースパケット番号で参照されるソースパケットまでのソースパケットの数 (SPN\_xxx) は、次式で算出される。

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}$$

図 4 9 に、SPN\_xxx が、5 である場合の例を示す。

## 【 0 1 7 4 】

TS\_recording\_rateは、24ビットの符号なし整数であり、この値は、DVRドライブ（書き込み部22）へまたはDVRドライブ（読み出し部28）からのAVストリームの必要な入出力のビットレートを与える。record\_time\_and\_dateは、Clipに対応するAVストリームが記録された時の日時をストアする56ビットのフィールドであり、年／月／日／時／分／秒について、14個の数字を4ビットのBinary Coded Decimal (BCD)で符号化したものである。例えば、2001/12/23:01:02:03は、"0x20011223010203"と符号化される。

## 【 0 1 7 5 】

durationは、Clipの総再生時間をアライバルタイムクロックに基づいた時間／分／秒の単位で示した24ビットのフィールドである。このフィールドは、6個の数字を4ビットのBinary Coded Decimal (BCD)で符号化したものである。例えば、01:45:30は、"0x014530"と符号化される。

## 【 0 1 7 6 】

time\_controlled\_flag:のフラグは、AVストリームファイルの記録モードを示す。このtime\_controlled\_flagが1である場合、記録モードは、記録してからの時間経過に対してファイルサイズが比例するようにして記録されるモードであることを示し、次式に示す条件を満たさなければならない。

$$\text{TS\_average\_rate} \times 192/188 \times (t - \text{start\_time}) - \alpha \leq \text{size\_clip}(t)$$

$$\leq \text{TS\_average\_rate} * 192 / 188 * (t - \text{start\_time}) + \alpha$$

ここで、TS\_average\_rateは、AVストリームファイルのトランスポートストリームの平均ビットレートをbytes/second の単位で表したものである。

## 【 0 1 7 7 】

また、上式において、t は、秒単位で表される時間を示し、start\_timeは、AVストリームファイルの最初のソースパケットが記録された時の時刻であり、秒単位で表される。size\_clip(t)は、時刻tにおけるAVストリームファイルのサイズをバイト単位で表したものであり、例えば、start\_timeから時刻tまでに10個のソースパケットが記録された場合、size\_clip(t)は10\*192バイトである。 $\alpha$ は、TS\_average\_rateに依存する定数である。

## 【 0 1 7 8 】

time\_controlled\_flagが0にセットされている場合、記録モードは、記録の時間経過とAVストリームのファイルサイズが比例するように制御していないことを示す。例えば、これは入力トランスポートストリームをトランスペアレント記録する場合である。

## 【 0 1 7 9 】

TS\_average\_rateは、time\_controlled\_flagが1にセットされている場合、この24ビットのフィールドは、上式で用いているTS\_average\_rateの値を示す。time\_controlled\_flagが0にセットされている場合、このフィールドは、何も意味を持たず、0にセットされなければならない。例えば、可変ビットレートのトランスポートストリームは、次に示す手順により符号化される。まずトランスポートレートをTS\_recording\_rateの値にセットする。次に、ビデオストリームを可変ビットレートで符号化する。そして、ヌルパケットを使用しない事によって、間欠的にトランスポートパケットを符号化する。

## 【 0 1 8 0 】

RSPN\_arrival\_time\_discontinuityの32ビットフィールドは、Bridge-Clip AV streamファイル上でアライバルタイムベースの不連続が発生する場所の相対アドレスである。RSPN\_arrival\_time\_discontinuityは、ソースパケット番号を単位とする大きさであり、Bridge-Clip AV streamファイルの最初のソースパケッ

トからClipInfo() において定義されるoffset\_SPNの値を初期値としてカウントされる。そのBridge-Clip AV streamファイルの中での絶対アドレスは、上述した

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}$$

に基づいて算出される。

#### 【 0 1 8 1 】

reserved\_for\_system\_useの144ビットのフィールドは、システム用にリザーブされている。is\_format\_identifier\_validのフラグが1である時、format\_identifierのフィールドが有効であることを示す。is\_original\_network\_ID\_validのフラグが1である場合、original\_network\_IDのフィールドが有効であることを示す。is\_transport\_stream\_ID\_validのフラグが1である場合、transport\_stream\_IDのフィールドが有効であることを示す。is\_service\_ID\_validのフラグが1である場合、service\_IDのフィールドが有効であることを示す。

#### 【 0 1 8 2 】

is\_country\_code\_validのフラグが1である時、country\_codeのフィールドが有効であることを示す。format\_identifierの32ビットフィールドは、トランスポートストリームの中でregistration\_descriptor (ISO/IEC13818-1で定義されている) が持つformat\_identifierの値を示す。original\_network\_IDの16ビットフィールドは、トランスポートストリームの中で定義されているoriginal\_network\_IDの値を示す。transport\_stream\_IDの16ビットフィールドは、トランスポートストリームの中で定義されているtransport\_stream\_IDの値を示す。

#### 【 0 1 8 3 】

service\_IDの16ビットフィールドは、トランスポートストリームの中で定義されているservice\_IDの値を示す。country\_codeの24ビットのフィールドは、ISO3166によって定義されるカンтриーコードを示す。それぞれのキャラクター文字は、ISO8859-1で符号化される。例えば、日本は"JPN"と表され、"0x4A 0x50 0x4E"と符号化される。stream\_format\_nameは、トランスポートストリームのストリーム定義をしているフォーマット機関の名称を示すISO-646の16個のキャラクターコードである。このフィールドの中の無効なバイトは、値'0xFF'がセッ



トされる。

【 0 1 8 4 】

format\_identifier、original\_network\_ID、transport\_stream\_ID、service\_ID、country\_code、およびstream\_format\_nameは、トランスポートストリームのサービスプロバイダを示すものであり、これにより、オーディオやビデオストリームの符号化制限、SI(サービスインフォメーション)の規格やオーディオビデオストリーム以外のプライベートデータストリームのストリーム定義を認識することができる。これらの情報は、デコーダが、そのストリームをデコードできるか否か、そしてデコードできる場合にデコード開始前にデコーダシステムの初期設定を行うために用いることが可能である。

【 0 1 8 5 】

次に、STC\_Infoについて説明する。ここでは、MPEG-2トランスポートストリームの中でSTCの不連続点(システムタイムベースの不連続点)を含まない時間区間をSTC\_sequenceと称し、Clipの中で、STC\_sequenceは、STC\_sequence\_idの値によって特定される。図50は、連続なSTC区間について説明する図である。同じSTC\_sequenceの中で同じSTCの値は、決して現れない(ただし、後述するように、Clipの最大時間長は制限されている)。従って、同じSTC\_sequenceの中で同じPTSの値もまた、決して現れない。AVストリームが、 $N(N>0)$ 個のSTC不連続点を含む場合、Clipのシステムタイムベースは、 $(N+1)$ 個のSTC\_sequenceに分割される。

【 0 1 8 6 】

STC\_Infoは、STCの不連続(システムタイムベースの不連続)が発生する場所のアドレスをストアする。図51を参照して説明するように、RSPN\_STC\_startが、そのアドレスを示し、最後のSTC\_sequenceを除く $k$ 番目( $k \geq 0$ )のSTC\_sequenceは、 $k$ 番目のRSPN\_STC\_startで参照されるソースパケットが到着した時刻から始まり、 $(k+1)$ 番目のRSPN\_STC\_startで参照されるソースパケットが到着した時刻で終わる。最後のSTC\_sequenceは、最後のRSPN\_STC\_startで参照されるソースパケットが到着した時刻から始まり、最後のソースパケットが到着した時刻で終了する。

## 【 0 1 8 7 】

図 5 2 は、STC\_Infoのシンタクスを示す図である。図 5 2 に示したSTC\_Infoのシンタクスについて説明するに、version\_numberは、このSTC\_Info()のバージョンナンバーを示す 4 個のキャラクター文字である。version\_numberは、ISO 646 に従って、"0045"と符号化されなければならない。

## 【 0 1 8 8 】

lengthは、このlengthフィールドの直後からSTC\_Info()の最後までSTC\_Info()のバイト数を示す 3 2 ビットの符号なし整数である。CPI()のCPI\_typeがTU\_map typeを示す場合、このlengthフィールドはゼロをセットしても良い。CPI()のCPI\_typeがEP\_map typeを示す場合、num\_of\_STC\_sequencesは1以上の値でなければならない。

## 【 0 1 8 9 】

num\_of\_STC\_sequencesの 8 ビットの符号なし整数は、Clipの中でのSTC\_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。所定のSTC\_sequenceに対応するSTC\_sequence\_idは、RSPN\_STC\_startを含むfor-loopの中で、そのSTC\_sequenceに対応するRSPN\_STC\_startの現れる順番により定義されるものである。STC\_sequence\_idは、0 から開始される。

## 【 0 1 9 0 】

RSPN\_STC\_startの 3 2 ビットフィールドは、AVストリームファイル上でSTC\_sequenceが開始するアドレスを示す。RSPN\_STC\_startは、AVストリームファイルの中でシステムタイムベースの不連続点が発生するアドレスを示す。RSPN\_STC\_startは、AVストリームの中で新しいシステムタイムベースの最初のPCRを持つソースパケットの相対アドレスとしても良い。RSPN\_STC\_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset\_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、既に上述した

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}$$

により算出される。

## 【 0 1 9 1 】

次に、図 4 5 に示した `zzzzz.clip` のシンタクス内の `ProgramInfo` について説明する。図 5 3 を参照しながら説明するに、ここでは、`Clip` の中で次の特徴をもつ時間区間を `program_sequence` と呼ぶ。まず、`PCR_PID` の値が変わらない。次に、ビデオエレメンタリーストリームの数が増加しない。また、それぞれのビデオストリームについての `PID` の値とその `VideoCodingInfo` によって定義される符号化情報が変化しない。さらに、オーディオエレメンタリーストリームの数が増加しない。また、それぞれのオーディオストリームについての `PID` の値とその `AudioCodingInfo` によって定義される符号化情報が変化しない。

## 【 0 1 9 2 】

`program_sequence` は、同一の時刻において、ただ 1 つのシステムタイムベースを持つ。`program_sequence` は、同一の時刻において、ただ 1 つの `PMT` を持つ。`ProgramInfo()` は、`program_sequence` が開始する場所のアドレスをストアする。`RSPN_program_sequence_start` が、そのアドレスを示す。

## 【 0 1 9 3 】

図 5 4 は、`ProgramInfo` のシンタクスを示す図である。図 5 4 に示した `ProgramInfo` のシンタクスを説明するに、`version_number` は、この `ProgramInfo()` のバージョンナンバーを示す 4 個のキャラクター文字である。`version_number` は、ISO 646 に従って、“0045” と符号化されなければならない。

## 【 0 1 9 4 】

`length` は、この `length` フィールドの直後から `ProgramInfo()` の最後までの `ProgramInfo()` のバイト数を示す 3 2 ビットの符号なし整数である。`CPI()` の `CPI_type` が `TU_map type` を示す場合、この `length` フィールドはゼロにセットされても良い。`CPI()` の `CPI_type` が `EP_map type` を示す場合、`number_of_programs` は 1 以上の値でなければならない。

## 【 0 1 9 5 】

`number_of_program_sequences` の 8 ビットの符号なし整数は、`Clip` の中での `program_sequence` の数を示す。この値は、このフィールドに続く `for-loop` のループ回数を示す。`Clip` の中で `program_sequence` が変化しない場合、`number_of_program_sequences` は 1 をセットされなければならない。`RSPN_program_sequence_start`

の 3 2 ビットフィールドは、AVストリームファイル上でプログラムシーケンスが開始する場所の相対アドレスである。

## 【 0 1 9 6 】

RSPN\_program\_sequence\_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset\_SPNの値を初期値としてカウントされる。そのAVストリームファイルの中での絶対アドレスは、

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}$$

により算出される。シンタクスのfor-loopの中でRSPN\_program\_sequence\_start値は、昇順に現れなければならない。

## 【 0 1 9 7 】

PCR\_PIDの 1 6 ビットフィールドは、そのprogram\_sequenceに有効なPCRフィールドを含むトランスポートパケットのPIDを示す。number\_of\_videosの 8 ビットフィールドは、video\_stream\_PIDとVideoCodingInfo()を含むfor-loopのループ回数を示す。number\_of\_audiosの 8 ビットフィールドは、audio\_stream\_PIDとAudioCodingInfo()を含むfor-loopのループ回数を示す。video\_stream\_PIDの 1 6 ビットフィールドは、そのprogram\_sequenceに有効なビデオストリームを含むトランスポートパケットのPIDを示す。このフィールドに続くVideoCodingInfo()は、そのvideo\_stream\_PIDで参照されるビデオストリームの内容を説明しなければならない。

## 【 0 1 9 8 】

audio\_stream\_PIDの 1 6 ビットフィールドは、そのprogram\_sequenceに有効なオーディオストリームを含むトランスポートパケットのPIDを示す。このフィールドに続くAudioCodingInfo()は、そのaudio\_stream\_PIDで参照されるビデオストリームの内容を説明しなければならない。

## 【 0 1 9 9 】

なお、シンタクスのfor-loopの中でvideo\_stream\_PIDの値の現れる順番は、そのprogram\_sequenceに有効なPMTの中でビデオストリームのPIDが符号化されている順番に等しくなければならない。また、シンタクスのfor-loopの中でaudio\_st

ream\_PIDの値の現れる順番は、そのprogram\_sequenceに有効なPMTの中でオーディオストリームのPIDが符号化されている順番に等しくなければならない。

#### 【 0 2 0 0 】

図 5 5 は、図 5 4 に示したProgramInfoのシンタクス内のVideoCodingInfoのシンタクスを示す図である。図 5 5 に示したVideoCodingInfoのシンタクスを説明するに、video\_formatの 8 ビットフィールドは、図 5 6 に示すように、ProgramInfo()の中のvideo\_stream\_PIDに対応するビデオフォーマットを示す。

#### 【 0 2 0 1 】

frame\_rateの 8 ビットフィールドは、図 5 7 に示すように、ProgramInfo()の中のvideo\_stream\_PIDに対応するビデオのフレームレートを示す。display\_aspect\_ratioの 8 ビットフィールドは、図 5 8 に示すように、ProgramInfo()の中のvideo\_stream\_PIDに対応するビデオの表示アスペクト比を示す。

#### 【 0 2 0 2 】

図 5 9 は、図 5 4 に示したProgramInfoのシンタクス内のAudioCodingInfoのシンタクスを示す図である。図 5 9 に示したAudioCodingInfoのシンタクスを説明するに、audio\_codingの 8 ビットフィールドは、図 6 0 に示すように、ProgramInfo()の中のaudio\_stream\_PIDに対応するオーディオの符号化方法を示す。

#### 【 0 2 0 3 】

audio\_component\_typeの 8 ビットフィールドは、図 6 1 に示すように、ProgramInfo()の中のaudio\_stream\_PIDに対応するオーディオのコンポーネントタイプを示す。sampling\_frequencyの 8 ビットフィールドは、図 6 2 に示すように、ProgramInfo()の中のaudio\_stream\_PIDに対応するオーディオのサンプリング周波数を示す。

#### 【 0 2 0 4 】

次に、図 4 5 に示したzzzzz.clipのシンタクス内のCPI (Characteristic Point Information)について説明する。CPIは、AVストリームの中の時間情報とそのファイルの中のアドレスとを関連づけるためにある。CPIには 2 つのタイプがあり、それらはEP\_mapとTU\_mapである。図 6 3 に示すように、CPI()の中のCPI\_typeがEP\_map typeの場合、そのCPI()はEP\_mapを含む。図 6 4 に示すように、CPI()

の中のCPI\_typeがTU\_map typeの場合、そのCPI()はTU\_mapを含む。1つのAVストリームは、1つのEP\_mapまたは一つのTU\_mapを持つ。AVストリームがSESFトランスポートストリームの場合、それに対応するClipはEP\_mapを持たなければならない。

#### 【0205】

図65は、CPIのシンタクスを示す図である。図65に示したCPIのシンタクスを説明するに、version\_numberは、このCPI()のバージョンナンバを示す4個のキャラクター文字である。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からCPI()の最後までCPI()のバイト数を示す32ビットの符号なし整数である。CPI\_typeは、図66に示すように、1ビットのフラグであり、ClipのCPIのタイプを表す。

#### 【0206】

次に、図65に示したCPIのシンタクス内のEP\_mapについて説明する。EP\_mapには、2つのタイプがあり、それはビデオストリーム用のEP\_mapとオーディオストリーム用のEP\_mapである。EP\_mapの中のEP\_map\_typeが、EP\_mapのタイプを区別する。Clipが1つ以上のビデオストリームを含む場合、ビデオストリーム用のEP\_mapが使用されなければならない。Clipがビデオストリームを含まず、1つ以上のオーディオストリームを含む場合、オーディオストリーム用のEP\_mapが使用されなければならない。

#### 【0207】

ビデオストリーム用のEP\_mapについて図67を参照して説明する。ビデオストリーム用のEP\_mapは、stream\_PID、PTS\_EP\_start、および、RSPN\_EP\_startというデータを持つ。stream\_PIDは、ビデオストリームを伝送するトランスポートパケットのPIDを示す。PTS\_EP\_startは、ビデオストリームのシーケンスヘッダから始めるアクセスユニットのPTSを示す。RSPN\_EP\_startは、AVストリームの中でPTS\_EP\_startにより参照されるアクセスユニットの第1バイト目を含むソースパケットのアドレスを示す。

#### 【0208】

EP\_map\_for\_one\_stream\_PID()と呼ばれるサブテーブルは、同じPIDを持つトランスポートパケットによって伝送されるビデオストリーム毎に作られる。Clipの中に複数のビデオストリームが存在する場合、EP\_mapは複数のEP\_map\_for\_one\_stream\_PID()を含んでも良い。

## 【 0 2 0 9 】

オーディオストリーム用のEP\_mapは、stream\_PID、PTS\_EP\_start、およびRSPN\_EP\_startというデータを持つ。stream\_PIDは、オーディオストリームを伝送するトランスポートパケットのPIDを示す。PTS\_EP\_startは、オーディオストリームのアクセスユニットのPTSを示す。RSPN\_EP\_startは、AVストリームの中でPTS\_EP\_startで参照されるアクセスユニットの第1バイト目を含むソースポケットのアドレスを示す。

## 【 0 2 1 0 】

EP\_map\_for\_one\_stream\_PID()と呼ばれるサブテーブルは、同じPIDを持つトランスポートパケットによって伝送されるオーディオストリーム毎に作られる。Clipの中に複数のオーディオストリームが存在する場合、EP\_mapは複数のEP\_map\_for\_one\_stream\_PID()を含んでも良い。

## 【 0 2 1 1 】

EP\_mapとSTC\_Infoの関係を説明するに、1つのEP\_map\_for\_one\_stream\_PID()は、STCの不連続点に関係なく1つのテーブルに作られる。RSPN\_EP\_startの値とSTC\_Info()において定義されるRSPN\_STC\_startの値を比較する事により、それぞれのSTC\_sequenceに属するEP\_mapのデータの境界が分かる（図68を参照）。

EP\_mapは、同じPIDで伝送される連続したストリームの範囲に対して、1つのEP\_map\_for\_one\_stream\_PIDを持たねばならない。図69に示したような場合、program#1とprogram#3は、同じビデオPIDを持つが、データ範囲が連続していないので、それぞれのプログラム毎にEP\_map\_for\_one\_stream\_PIDを持たねばならない。

## 【 0 2 1 2 】

図70は、EP\_mapのシンタクスを示す図である。図70に示したEP\_mapのシンタクスを説明するに、EP\_typeは、4ビットのフィールドであり、図71に示す

ように、EP\_mapのエントリーポイントタイプを示す。EP\_typeは、このフィールドに続くデータフィールドのセマンティクスを示す。Clipが1つ以上のビデオストリームを含む場合、EP\_typeは0('video')にセットされなければならない。または、Clipがビデオストリームを含まず、1つ以上のオーディオストリームを含む場合、EP\_typeは1('audio')にセットされなければならない。

## 【 0 2 1 3 】

number\_of\_stream\_PIDsの16ビットのフィールドは、EP\_map()の中のnumber\_of\_stream\_PIDsを変数にもつfor-loopのループ回数を示す。stream\_PID(k)の16ビットのフィールドは、EP\_map\_for\_one\_stream\_PID(num\_EP\_entries(k))によって参照されるk番目のエレメンタリストリーム（ビデオまたはオーディオストリーム）を伝送するトランスポートパケットのPIDを示す。EP\_typeが0('video')に等しい場合、そのエレメンタリストリームはビデオストリームでなければならない。また、EP\_typeが1('audio')に等しい場合、そのエレメンタリストリームはオーディオストリームでなければならない。

## 【 0 2 1 4 】

num\_EP\_entries(k)の16ビットのフィールドは、EP\_map\_for\_one\_stream\_PID(num\_EP\_entries(k))によって参照されるnum\_EP\_entries(k)を示す。EP\_map\_for\_one\_stream\_PID\_Start\_address(k)：この32ビットのフィールドは、EP\_map()の中でEP\_map\_for\_one\_stream\_PID(num\_EP\_entries(k))が始まる相対バイト位置を示す。この値は、EP\_map()の第1バイト目からの大きさで示される。

## 【 0 2 1 5 】

padding\_wordは、EP\_map()のシンタクスにしたがって挿入されなければならない。XとYは、ゼロまたは任意の正の整数でなければならない。それぞれのパディングワードは、任意の値を取っても良い。

## 【 0 2 1 6 】

図72は、EP\_map\_for\_one\_stream\_PIDのシンタクスを示す図である。図72に示したEP\_map\_for\_one\_stream\_PIDのシンタクスを説明するに、PTS\_EP\_startの32ビットのフィールドのセマンティクスは、EP\_map()において定義されるEP\_typeにより異なる。EP\_typeが0('video')に等しい場合、このフィールドは、



ビデオストリームのシーケンスヘッダで始まるアクセスユニットの33ビット精度のPTSの上位32ビットを持つ。EP\_typeが1 ('audio')に等しい場合、このフィールドは、オーディオストリームのアクセスユニットの33ビット精度のPTSの上位32ビットを持つ。

## 【0217】

RSPN\_EP\_startの32ビットのフィールドのセマンティクスは、EP\_map()において定義されるEP\_typeにより異なる。EP\_typeが0 ('video')に等しい場合、このフィールドは、AVストリームの中でPTS\_EP\_startにより参照されるアクセスユニットのシーケンスヘッダの第1バイト目を含むソースポケットの相対アドレスを示す。または、EP\_typeが1 ('audio')に等しい場合、このフィールドは、AVストリームの中でPTS\_EP\_startにより参照されるアクセスユニットのオーディオフレームの第1バイト目を含むソースポケットの相対アドレスを示す。

## 【0218】

RSPN\_EP\_startは、ソースポケット番号を単位とする大きさであり、AVストリームファイルの最初のソースポケットからClipInfo()において定義されるoffset\_SPNの値を初期値としてカウントされる。そのAVストリームファイルの中での絶対アドレスは、

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}$$

により算出される。シンタクスのfor-loopの中でRSPN\_EP\_startの値は、昇順に現れなければならない。

## 【0219】

次に、TU\_mapについて、図73を参照して説明する。TU\_mapは、ソースポケットのアライバルタイムクロック（到着時刻ベースの時計）に基づいて、1つの時間軸を作る。その時間軸は、TU\_map\_time\_axisと呼ばれる。TU\_map\_time\_axisの原点は、TU\_map()の中のoffset\_timeによって示される。TU\_map\_time\_axisは、offset\_timeから一定の単位に分割される。その単位を、time\_unitと称する。

## 【0220】

AVストリームの中の各々のtime\_unitの中で、最初の完全な形のソースポケットのAVストリームファイル上のアドレスが、TU\_mapにストアされる。これらのア

ドレスを、RSPN\_time\_unit\_startと称する。TU\_map\_time\_axis上において、 $k$  ( $k \geq 0$ ) 番目のtime\_unitが始まる時刻は、TU\_start\_time( $k$ )と呼ばれる。この値は次式に基づいて算出される。

$$\text{TU\_start\_time}(k) = \text{offset\_time} + k * \text{time\_unit\_size}$$

TU\_start\_time( $k$ )は、45kHzの精度を持つ。

#### 【 0 2 2 1 】

図 7 5 は、TU\_mapのシンタクスを示す図である。図 7 5 に示したTU\_mapのシンタクスを説明するに、offset\_timeの32bit長のフィールドは、TU\_map\_time\_axisに対するオフセットタイムを与える。この値は、Clipの中の最初のtime\_unitに対するオフセット時刻を示す。offset\_timeは、27MHz精度のアライバルタイムクロックから導き出される45kHzクロックを単位とする大きさである。AVストリームが新しいClipとして記録される場合、offset\_timeはゼロにセットされなければならない。

#### 【 0 2 2 2 】

time\_unit\_sizeの32ビットフィールドは、time\_unitの大きさを与えるものであり、それは27MHz精度のアライバルタイムクロックから導き出される45kHzクロックを単位とする大きさである。time\_unit\_sizeは、1秒以下 ( $\text{time\_unit\_size} \leq 4500$ ) にすることが良い。number\_of\_time\_unit\_entriesの32ビットフィールドは、TU\_map()の中にストアされているtime\_unitのエントリー数を示す。

#### 【 0 2 2 3 】

RSPN\_time\_unit\_startの32ビットフィールドは、AVストリームの中でそれぞれのtime\_unitが開始する場所の相対アドレスを示す。RSPN\_time\_unit\_startは、ソースパケット番号を単位とする大きさであり、AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset\_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}$$

により算出される。シンタクスのfor-loopの中でRSPN\_time\_unit\_startの値は、昇順に現れなければならない。(k+1)番目のtime\_unitの中にソースパケットが何

もない場合、(k+1)番目のRSPN\_time\_unit\_startは、k番目のRSPN\_time\_unit\_startと等しくなければならない。

#### 【 0 2 2 4 】

図 4 5 に示したzzzzz.clipのシンタクス内のClipMarkについて説明する。ClipMarkは、クリップについてのマーク情報であり、ClipMarkの中にストアされる。このマークは、記録器（記録再生装置1）によってセットされるものであり、ユーザによってセットされるものではない。

#### 【 0 2 2 5 】

図 7 5 は、ClipMarkのシンタクスを示す図である。図 7 5 に示したClipMarkのシンタクスを説明するに、version\_numberは、このClipMark()のバージョンナンバーを示す4個のキャラクター文字である。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。

#### 【 0 2 2 6 】

lengthは、このlengthフィールドの直後からClipMark()の最後までのClipMark()のバイト数を示す32ビットの符号なし整数である。number\_of\_clip\_marksは、ClipMarkの中にストアされているマークの個数を示す16ビットの符号なし整数。number\_of\_clip\_marksは、0であってもよい。mark\_typeは、マークのタイプを示す8ビットのフィールドであり、図 7 6 に示すテーブルに従って符号化される。

#### 【 0 2 2 7 】

mark\_time\_stampは、32ビットフィールドであり、マークが指定されたポイントを示すタイムスタンプをストアする。mark\_time\_stampのセマンティクスは、図 7 7 に示すように、PlayList()の中のCPI\_typeにより異なる。

#### 【 0 2 2 8 】

STC\_sequence\_idは、CPI()の中のCPI\_typeがEP\_map typeを示す場合、この8ビットのフィールドは、マークが置かれているところのSTC連続区間のSTC\_sequence\_idを示す。CPI()の中のCPI\_typeがTU\_map typeを示す場合、この8ビットのフィールドは何も意味を持たず、ゼロにセットされる。character\_setの8ビットのフィールドは、mark\_nameフィールドに符号化されているキャラクター文字

の符号化方法を示す。その符号化方法は、図19に示される値に対応する。

#### 【0229】

name\_lengthの8ビットフィールドは、Mark\_nameフィールドの中に示されるマーク名のバイト長を示す。mark\_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname\_length数のバイト数が、有効なキャラクター文字であり、それはマークの名称を示す。mark\_nameフィールドの中で、それら有効なキャラクター文字の後の値は、どんな値が入っていても良い。

#### 【0230】

ref\_thumbnail\_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref\_thumbnail\_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref\_thumbnail\_indexの値を用いて参照される。ref\_thumbnail\_indexフィールドが、0xFFFFである場合、そのマークにはサムネイル画像が付加されていない。

#### 【0231】

MakersPrivateDataについては、図22を参照して既に説明したので、その説明は省略する。

#### 【0232】

次に、サムネイルインフォメーション (Thumbnail Information) について説明する。サムネイル画像は、menu.thmbファイルまたはmark.thmbファイルにストアされる。これらのファイルは同じシンタクス構造であり、ただ1つのThumbnail()を持つ。menu.thmbファイルは、メニューサムネイル画像、すなわちVolumeを代表する画像、および、それぞれのPlaylistを代表する画像をストアする。すべてのメニューサムネイルは、ただ1つのmenu.thmbファイルにストアされる。

#### 【0233】

mark.thmbファイルは、マークサムネイル画像、すなわちマーク点を表すピクチャをストアする。すべてのPlaylistおよびClipに対するすべてのマークサムネイルは、ただ1つのmark.thmbファイルにストアされる。サムネイルは頻繁に追加、削除されるので、追加操作と部分削除の操作は容易に高速に実行できなけれ

ばならない。この理由のため、Thumbnail()はブロック構造を有する。画像のデータはいくつかの部分に分割され、各部分の一つのtn\_blockに格納される。1つの画像データは連続したtn\_blockに格納される。tn\_blockの列には、使用されていないtn\_blockが存在してもよい。1つのサムネイル画像のバイト長は可変である。

#### 【 0 2 3 4 】

図 7 8 は、menu.thmbとmark.thmbのシンタクスを示す図であり、図 7 9 は、図 7 8 に示したmenu.thmbとmark.thmbのシンタクス内のThumbnailのシンタクスを示す図である。図 7 9 に示したThumbnailのシンタクスについて説明するに、version\_numberは、このThumbnail()のバージョンナンバーを示す4個のキャラクター文字である。version\_numberは、ISO 646に従って、“0045”と符号化されなければならない。

#### 【 0 2 3 5 】

lengthは、このlengthフィールドの直後からThumbnail()の最後までMakersPrivateData()のバイト数を示す32ビットの符号なし整数である。tn\_blocks\_start\_addressは、Thumbnail()の先頭のバイトからの相対バイト数を単位として、最初のtn\_blockの先頭バイトアドレスを示す32ビットの符号なし整数である。相対バイト数はゼロからカウントされる。number\_of\_thumbnailsは、Thumbnail()の中に含まれているサムネイル画像のエントリー数を与える16ビットの符号なし整数である。

#### 【 0 2 3 6 】

tn\_block\_sizeは、1024バイトを単位として、1つのtn\_blockの大きさを与える16ビットの符号なし整数である。例えば、tn\_block\_size=1ならば、それは1つのtn\_blockの大きさが1024バイトであることを示す。number\_of\_tn\_blocksは、このThumbnail()中のtn\_blockのエントリー数を表す16ビットの符号なし整数である。thumbnail\_indexは、このthumbnail\_indexフィールドから始まるforループ一回分のサムネイル情報で表されるサムネイル画像のインデックス番号を表す16ビットの符号なし整数である。thumbnail\_indexとして、0xFFFFという値を使用してはならない。thumbnail\_indexはUIAppInfoVolume()、UIAppInfoPlay

List()、PlayListMark()、およびClipMark()の中のref\_thumbnail\_indexによって参照される。

#### 【0237】

thumbnail\_picture\_formatは、サムネイル画像のピクチャフォーマットを表す8ビットの符号なし整数で、図80に示すような値をとる。表中のDCFとPNGは”menu.thmb”内でのみ許される。マークサムネイルは、値”0x00”(MPEG-2 Video I-picture)をとらなければならない。

#### 【0238】

picture\_data\_sizeは、サムネイル画像のバイト長をバイト単位で示す32ビットの符号なし整数である。start\_tn\_block\_numberは、サムネイル画像のデータが始まるtn\_blockのtn\_block番号を表す16ビットの符号なし整数である。サムネイル画像データの先頭は、tb\_blockの先頭と一致していなければならない。tn\_block番号は、0から始まり、tn\_blockのfor-ループ中の変数kの値に関係する。

#### 【0239】

x\_picture\_lengthは、サムネイル画像のフレーム画枠の水平方向のピクセル数を表す16ビットの符号なし整数である。y\_picture\_lengthは、サムネイル画像のフレーム画枠の垂直方向のピクセル数を表す16ビットの符号なし整数である。tn\_blockは、サムネイル画像がストアされる領域である。Thumbnail()の中のすべてのtn\_blockは、同じサイズ(固定長)であり、その大きさはtn\_block\_sizeによって定義される。

#### 【0240】

図81は、サムネイル画像データがどのようにtn\_blockに格納されるかを模式的に表した図である。図81のように、各サムネイル画像データはtn\_blockの先頭から始まり、1 tn\_blockを超える大きさの場合は、連続する次のtn\_blockを使用してストアされる。このようにすることにより、可変長であるピクチャデータが、固定長のデータとして管理することが可能となり、削除といった編集に対して簡便な処理により対応する事ができるようになる。

#### 【0241】

次に、AVストリームファイルについて説明する。AVストリームファイルは、“M2TS”ディレクトリ（図14）にストアされる。AVストリームファイルには、2つのタイプがあり、それらは、Clip AVストリームとBridge-Clip AVストリームファイルである。両方のAVストリーム共に、これ以降で定義されるDVR MPEG-2トランスポートストリームファイルの構造でなければならない。

#### 【0242】

まず、DVR MPEG-2 トランスポートストリームについて説明する。DVR MPEG-2 トランスポートストリームの構造は、図82に示すようになっている。AVストリームファイルは、DVR MPEG2トランスポートストリームの構造を持つ。DVR MPEG2 トランスポートストリームは、整数個のAligned unitから構成される。Aligned unitの大きさは、6144 バイト (2048\*3 バイト)である。Aligned unitは、ソースパケットの第1バイト目から始まる。ソースパケットは、192バイト長である。一つのソースパケットは、TP\_extra\_headerとトランスポートパケットから成る。TP\_extra\_headerは、4 バイト長であり、またトランスポートパケットは、188バイト長である。

#### 【0243】

1つのAligned unitは、32個のソースパケットから成る。DVR MPEG2 トランスポートストリームの中の最後のAligned unitも、また32個のソースパケットから成る。よって、DVR MPEG2トランスポートストリームは、Aligned unitの境界で終端する。ディスクに記録される入力トランスポートストリームのトランスポートパケットの数が32の倍数でない時、ヌルパケット (PID=0x1FFFのトランスポートパケット) を持ったソースパケットを最後のAligned unitに使用しなければならない。ファイルシステムは、DVR MPEG2 トランスポートストリームに余分な情報を付加してはならない。

#### 【0244】

図83に、DVR MPEG-2トランスポートストリームのレコーダモデルを示す。図83に示したレコーダは、レコーディングプロセスを規定するための概念上のモデルである。DVR MPEG-2トランスポートストリームは、このモデルに従う。

#### 【0245】

MPEG-2トランスポートストリームの入力タイミングについて説明する。入力MP EG2トランスポートストリームは、フルトランスポートストリームまたはパシ ャルトランスポートストリームである。入力されるMPEG2トランスポートストリ ームは、ISO/IEC13818-1 ([2])またはISO/IEC13818-9 ([5])に従っていなければ ならない。MPEG2トランスポートストリームの*i*番目のバイトは、T-STD(ISO/IEC 13818-1で規定されるTransport stream system target decoder)とソースパケッ タイザーへ、時刻*t(i)*に同時に入力される。Rpkは、トランスポートパケットの 入力レートの瞬時的な最大値である。

## 【 0 2 4 6 】

2 7 MHz PLL 5 2 は、2 7 MHzクロックの周波数を発生する。2 7 MHzクロック の周波数は、MPEG-2トランスポートストリームのPCR (Program Clock Reference )の値にロックされる。arrival time clock counter 5 3 は、2 7 MHzの周波数の パルスをカウントするバイナリーカウンタである。Arrival\_time\_clock(*i*)は 、時刻*t(i)*におけるArrival time clock counterのカウント値である。

## 【 0 2 4 7 】

source packetizer 5 4 は、すべてのトランスポートパケットにTP\_extra\_head erを付加し、ソースパケットを作る。Arrival\_time\_stampは、トランスポートパ ケットの第1バイト目がT-STDとソースパケッタイザーの両方へ到着する時刻を 表す。Arrival\_time\_stamp(*k*)は、次式で示されるようにArrival\_time\_clock(*k*) のサンプル値であり、ここで、*k*はトランスポートパケットの第1バイト目を示 す。

$$\text{arrival\_time\_stamp}(k) = \text{arrival\_time\_clock}(k) \% 2^{30}$$

## 【 0 2 4 8 】

2つの連続して入力されるトランスポートパケットの時間間隔が、 $2^{30}/270000$  00秒(約40秒)以上になる場合、その2つのトランスポートパケットのarrival\_ time\_stampの差分は、 $2^{30}/27000000$ 秒になるようにセットされるべきである。レ コーダは、そのようになる場合に備えてある。

## 【 0 2 4 9 】

smoothing buffer 5 5 は、入力トランスポートストリームのビットレートをス



ムージングする。スムージングバッファは、オーバーフローしてはならない。Rmaxは、スムージングバッファが空でない時のスムージングバッファからのソースパケットの出力ビットレートである。スムージングバッファが空である時、スムージングバッファからの出力ビットレートはゼロである。

## 【0250】

次に、DVR MPEG-2トランスポートストリームのレコーダモデルのパラメータについて説明する。Rmaxという値は、AVストリームファイルに対応するClipInfo()において定義されるTS\_recording\_rateによって与えられる。この値は、次式により算出される。

$$R_{\max} = \text{TS\_recording\_rate} * 192/188$$

TS\_recording\_rateの値は、bytes/secondを単位とする大きさである。

## 【0251】

入力トランスポートストリームがSESFトランスポートストリームの場合、Rpkは、AVストリームファイルに対応するClipInfo()において定義されるTS\_recording\_rateに等しくなければならない。入力トランスポートストリームがSESFトランスポートストリームでない場合、この値はMPEG-2 transport streamのデスクリプター、例えばmaximum\_bitrate\_descriptorやpartial\_transport\_stream\_descriptorなど、において定義される値を参照しても良い。

## 【0252】

smoothing buffer sizeは、入力トランスポートストリームがSESFトランスポートストリームの場合、スムージングバッファの大きさはゼロである。入力トランスポートストリームがSESFトランスポートストリームでない場合、スムージングバッファの大きさはMPEG-2 transport streamのデスクリプター、例えばsmoothing\_buffer\_descriptor、short\_smoothing\_buffer\_descriptor、partial\_transport\_stream\_descriptorなどにおいて定義される値を参照しても良い。

## 【0253】

記録機（レコーダ）および再生機（プレーヤ）は、十分なサイズのバッファを用意しなければならない。デフォルトのバッファサイズは、1536 bytes である。

## 【 0 2 5 4 】

次に、DVR MPEG-2トランスポートストリームのプレーヤモデルについて説明する。図 8 4 は、DVR MPEG-2トランスポートストリームのプレーヤモデルを示す図である。これは、再生プロセスを規定するための概念上のモデルである。DVR MPEG-2トランスポートストリームは、このモデルに従う。

## 【 0 2 5 5 】

27MHz X-tal 6 1 は、2 7 Mhzの周波数を発生する。2 7 MHz周波数の誤差範囲は、 $\pm 30$  ppm (27000000  $\pm$  810 Hz)でなければならない。arrival time clock counter 6 2 は、2 7 MHzの周波数のパルスをカウントするバイナリーカウンタである。Arrival\_time\_clock(i)は、時刻t(i)におけるArrival time clock counterのカウント値である。

## 【 0 2 5 6 】

smoothing buffer 6 4 において、Rmaxは、スムージングバッファがフルでない時のスムージングバッファへのソースパケットの入力ビットレートである。スムージングバッファがフルである時、スムージングバッファへの入力ビットレートはゼロである。

## 【 0 2 5 7 】

MPEG-2トランスポートストリームの出力タイミングを説明するに、現在のソースパケットのarrival\_time\_stampがarrival\_time\_clock(i)のLSB 3 0 ビットの値と等しい時、そのソースパケットのトランスポートパケットは、スムージングバッファから引き抜かれる。Rpkは、トランスポートパケットレートの瞬時的な最大値である。スムージングバッファは、アンダーフローしてはならない。

## 【 0 2 5 8 】

DVR MPEG-2トランスポートストリームのプレーヤモデルのパラメータについては、上述したDVR MPEG-2トランスポートストリームのレコーダモデルのパラメータと同一である。

## 【 0 2 5 9 】

図 8 5 は、Source packetのシンタクスを示す図である。transport\_packet()は、ISO/IEC 13818-1で規定されるMPEG-2トランスポートパケットである。図 8

5に示したSource packetのシンタクス内のTP\_Extra\_headerのシンタクスを図86に示す。図86に示したTP\_Extra\_headerのシンタクスについて説明するに、copy\_permission\_indicatorは、トランスポートパケットのペイロードのコピー制限を表す整数である。コピー制限は、copy free、no more copy、copy once、またはcopy prohibitedとすることができる。図87は、copy\_permission\_indicatorの値と、それらによって指定されるモードの関係を示す。

## 【0260】

copy\_permission\_indicatorは、すべてのトランスポートパケットに付加される。IEEE1394デジタルインターフェースを使用して入力トランスポートストリームを記録する場合、copy\_permission\_indicatorの値は、IEEE1394 isochronous packet headerの中のEMI (Encryption Mode Indicator)の値に関連付けても良い。IEEE1394デジタルインターフェースを使用しないで入力トランスポートストリームを記録する場合、copy\_permission\_indicatorの値は、トランスポートパケットの中に埋め込まれたCCIの値に関連付けても良い。アナログ信号入力をセルフエンコードする場合、copy\_permission\_indicatorの値は、アナログ信号のCGMS-Aの値に関連付けても良い。

## 【0261】

arrival\_time\_stampは、次式

$$\text{arrival\_time\_stamp}(k) = \text{arrival\_time\_clock}(k) \% 2^{30}$$

において、arrival\_time\_stampによって指定される値を持つ整数値である。

## 【0262】

Clip AVストリームの定義をするに、Clip AVストリームは、上述したような定義がされるDVR MPEG-2トランスポートストリームの構造を持たねばならない。arrival\_time\_clock(i)は、Clip AVストリームの中で連続して増加しなければならない。Clip AVストリームの中にシステムタイムベース (STCベース) の不連続点が存在したとしても、そのClip AVストリームのarrival\_time\_clock(i)は、連続して増加しなければならない。

## 【0263】

Clip AVストリームの中の開始と終了の間のarrival\_time\_clock(i)の差分の最

大値は、26時間であればならない。この制限は、MPEG2トランスポートストリームの中にシステムタイムベース（STCベース）の不連続点が存在しない場合に、Clip AVストリームの中で同じ値のPTS(Presentation Time Stamp)が決して現れないことを保証する。MPEG2システムズ規格は、PTSのラップアラウンド周期を233/90000秒(約26.5時間).と規定している。

## 【0264】

Bridge-Clip AVストリームの定義をするに、Bridge-Clip AVストリームは、上述したような定義がされるDVR MPEG-2トランスポートストリームの構造を持たねばならない。Bridge-Clip AVストリームは、1つのアライバルタイムベースの不連続点を含まなければならない。アライバルタイムベースの不連続点の前後のトランスポートストリームは、後述する符号化の制限に従わなければならない、かつ後述するDVR-STDに従わなければならない。

## 【0265】

本実施の形態においては、編集におけるPlayItem間のビデオとオーディオのシームレス接続をサポートする。PlayItem間をシームレス接続にすることは、プレーヤ/レコーダに”データの連続供給”と”シームレスな復号処理”を保証する。”データの連続供給”とは、ファイルシステムが、デコーダにバッファのアンダーフローを起こさせる事のないように必要なビットレートでデータを供給する事を保証できることである。データのリアルタイム性を保証して、データをディスクから読み出すことができるように、データが十分な大きさの連続したブロック単位でストアされるようにする。

## 【0266】

”シームレスな復号処理”とは、プレーヤが、デコーダの再生出力にポーズやギャップを起こさせる事なく、ディスクに記録されたオーディオビデオデータを表示できることである。

## 【0267】

シームレス接続されているPlayItemが参照するAVストリームについて説明する。先行するPlayItemと現在のPlayItemの接続が、シームレス表示できるように保証されているかどうかは、現在のPlayItemにおいて定義されているconnection\_c

onditionフィールドから判断することができる。PlayItem間のシームレス接続は、Bridge-Clipを使用する方法と使用しない方法がある。

#### 【 0 2 6 8 】

図 8 8 は、Bridge-Clipを使用する場合の先行するPlayItemと現在のPlayItemの関係を示している。図 8 8 においては、プレーヤが読み出すストリームデータが、影をつけて示されている。図 8 8 に示したTS1は、Clip1 (Clip AVストリーム) の影を付けられたストリームデータとBridge-ClipのRSPN\_arrival\_time\_discontinuityより前の影を付けられたストリームデータから成る。

#### 【 0 2 6 9 】

TS1のClip1の影を付けられたストリームデータは、先行するPlayItemのIN\_time (図 8 8 においてIN\_time1で図示されている) に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスから、RSPN\_exit\_from\_previous\_Clipで参照されるソースパケットまでのストリームデータである。TS1に含まれるBridge-ClipのRSPN\_arrival\_time\_discontinuityより前の影を付けられたストリームデータは、Bridge-Clipの最初のソースパケットから、RSPN\_arrival\_time\_discontinuityで参照されるソースパケットの直前のソースパケットまでのストリームデータである。

#### 【 0 2 7 0 】

また、図 8 8 におけるTS2は、Clip2 (Clip AVストリーム) の影を付けられたストリームデータとBridge-ClipのRSPN\_arrival\_time\_discontinuity以後の影を付けられたストリームデータから成る。TS2に含まれるBridge-ClipのRSPN\_arrival\_time\_discontinuity以後の影を付けられたストリームデータは、RSPN\_arrival\_time\_discontinuityで参照されるソースパケットから、Bridge-Clipの最後のソースパケットまでのストリームデータである。TS2のClip2の影を付けられたストリームデータは、RSPN\_enter\_to\_current\_Clipで参照されるソースパケットから、現在のPlayItemのOUT\_time (図 8 8 においてOUT\_time2で図示されている) に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスまでのストリームデータである。

#### 【 0 2 7 1 】

図 8 9 は、Bridge-Clipを使用しない場合の先行するPlayItemと現在のPlayItemの関係を示している。この場合、プレーヤが読み出すストリームデータは、影をつけて示されている。図 8 9 におけるTS1は、Clip1 (Clip AVストリーム)の影を付けられたストリームデータから成る。TS1のClip1の影を付けられたストリームデータは、先行するPlayItemのIN\_time (図 8 9 においてIN\_time1で図示されている) に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスから始まり、Clip1の最後のソースパケットまでのデータである。また、図 8 9 におけるTS2は、Clip2 (Clip AVストリーム)の影を付けられたストリームデータから成る。

#### 【 0 2 7 2 】

TS2のClip2の影を付けられたストリームデータは、Clip2の最初のソースパケットから始まり、現在のPlayItemのOUT\_time (図 8 9 においてOUT\_time2で図示されている) に対応するプレゼンテーションユニットを復号する為に必要なストリームのアドレスまでのストリームデータである。

#### 【 0 2 7 3 】

図 8 8 と図 8 9 において、TS1とTS2は、ソースパケットの連続したストリームである。次に、TS1とTS2のストリーム規定と、それらの間の接続条件について考える。まず、シームレス接続のための符号化制限について考える。トランスポートストリームの符号化構造の制限として、まず、TS1とTS2の中に含まれるプログラムの数は、1 でなければならない。TS1とTS2の中に含まれるビデオストリームの数は、1 でなければならない。TS1とTS2の中に含まれるオーディオストリームの数は、2 以下でなければならない。TS1とTS2の中に含まれるオーディオストリームの数は、等しくなければならない。TS1および/またはTS2の中に、上記以外のエレメンタリーストリームまたはプライベートストリームが含まれていても良い。

#### 【 0 2 7 4 】

ビデオビットストリームの制限について説明する。図 9 0 は、ピクチャの表示順序で示すシームレス接続の例を示す図である。接続点においてビデオストリームをシームレスに表示できるためには、OUT\_time1 (Clip1のOUT\_time) の後とIN

\_time2 (Clip2のIN\_time) の前に表示される不必要なピクチャは、接続点付近のClipの部分的なストリームを再エンコードするプロセスにより、除去されなければならない。

【 0 2 7 5 】

図 9 0 に示したような場合において、BridgeSequenceを使用してシームレス接続を実現する例を、図 9 1 に示す。RSPN\_arrival\_time\_discontinuityより前のBridge-Clipのビデオストリームは、図 9 0 のClip1のOUT\_time1に対応するピクチャまでの符号化ビデオストリームから成る。そして、そのビデオストリームは先行するClip1のビデオストリームに接続され、1つの連続でMPEG 2 規格に従ったエレメンタリーストリームとなるように再エンコードされている。

【 0 2 7 6 】

同様に、RSPN\_arrival\_time\_discontinuity以後のBridge-Clipのビデオストリームは、図 9 0 のClip2のIN\_time2に対応するピクチャ以後の符号化ビデオストリームから成る。そして、そのビデオストリームは、正しくデコード開始する事ができて、これに続くClip2のビデオストリームに接続され、1つの連続でMPEG2規格に従ったエレメンタリーストリームとなるように再エンコードされている。Bridge-Clipを作るためには、一般に、数枚のピクチャは再エンコードしなければならない、それ以外のピクチャはオリジナルのClipからコピーすることができる。

【 0 2 7 7 】

図 9 0 に示した例の場合にBridgeSequenceを使用しないでシームレス接続を実現する例を図 9 2 に示す。Clip1のビデオストリームは、図 9 0 のOUT\_time1に対応するピクチャまでの符号化ビデオストリームから成り、それは、1つの連続でMPEG 2 規格に従ったエレメンタリーストリームとなるように再エンコードされている。同様に、Clip2のビデオストリームは、図 9 0 のClip2のIN\_time2に対応するピクチャ以後の符号化ビデオストリームから成り、それは、一つの連続でMPEG 2 規格に従ったエレメンタリーストリームとなるように再エンコードされている。

【 0 2 7 8 】

ビデオストリームの符号化制限について説明するに、まず、TS1とTS2のビデオストリームのフレームレートは、等しくなければならない。TS1のビデオストリームは、`sequence_end_code`で終端しなければならない。TS2のビデオストリームは、Sequence Header、GOP Header、そしてI-ピクチャで開始しなければならない。TS2のビデオストリームは、クローズドGOPで開始しなければならない。

## 【 0 2 7 9 】

ビットストリームの中で定義されるビデオプレゼンテーションユニット（フレームまたはフィールド）は、接続点を挟んで連続でなければならない。接続点において、フレームまたはフィールドのギャップがあってはならない。接続点において、トップボトムのフィールドシーケンスは連続でなければならない。3-2プルダウンを使用するエンコードの場合は、“`top_field_first`” および “`repeat_first_field`” フラグを書き換える必要があるかもしれない、またはフィールドギャップの発生を防ぐために局所的に再エンコードするようにしても良い。

## 【 0 2 8 0 】

オーディオビットストリームの符号化制限について説明するに、TS1とTS2のオーディオのサンプリング周波数は、同じでなければならない。TS1とTS2のオーディオの符号化方法（例、MPEG1レイヤ2、AC-3、SESF LPCM、AAC）は、同じでなければならない。

## 【 0 2 8 1 】

次に、MPEG-2トランスポートストリームの符号化制限について説明するに、TS1のオーディオストリームの最後のオーディオフレームは、TS1の最後の表示ピクチャの表示終了時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。TS2のオーディオストリームの最初のオーディオフレームは、TS2の最初の表示ピクチャの表示開始時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。

## 【 0 2 8 2 】

接続点において、オーディオプレゼンテーションユニットのシーケンスにギャップがあってはならない。図9 3に示すように、2オーディオフレーム区間未満のオーディオプレゼンテーションユニットの長さで定義されるオーバーラップが



あっても良い。TS2のエレメンタリーストリームを伝送する最初の packets は、ビデオ packets でなければならない。接続点におけるトランスポートストリームは、後述するDVR-STDに従わなくてはならない。

#### 【 0 2 8 3 】

ClipおよびBridge-Clipの制限について説明するに、TS1とTS2は、それぞれの中にアライバルタイムベースの不連続点を含んではならない。

#### 【 0 2 8 4 】

以下の制限は、Bridge-Clipを使用する場合にのみ適用される。TS1の最後のソース packets とTS2の最初のソース packets の接続点においてのみ、Bridge-Clip AVストリームは、ただ1つのアライバルタイムベースの不連続点を持つ。ClipInfo()において定義されるRSPN\_arrival\_time\_discontinuityが、その不連続点のアドレスを示し、それはTS2の最初のソース packets を参照するアドレスを示さなければならない。

#### 【 0 2 8 5 】

BridgeSequenceInfo()において定義されるRSPN\_exit\_from\_previous\_Clipによって参照されるソース packets は、Clip1の中のどのソース packets でも良い。それは、Aligned unitの境界である必要はない。BridgeSequenceInfo()において定義されるRSPN\_enter\_to\_current\_Clipによって参照されるソース packets は、Clip2の中のどのソース packets でも良い。それは、Aligned unitの境界である必要はない。

#### 【 0 2 8 6 】

PlayItemの制限について説明するに、先行するPlayItemのOUT\_time (図 8 8、図 8 9 において示されるOUT\_time1) は、TS1の最後のビデオプレゼンテーションユニットの表示終了時刻を示さなければならない。現在のPlayItemのIN\_time (図 8 8、図 8 9 において示されるIN\_time2) は、TS2の最初のビデオプレゼンテーションユニットの表示開始時刻を示さなければならない。

#### 【 0 2 8 7 】

Bridge-Clipを使用する場合のデータアロケーションの制限について、図 9 4 を参照して説明するに、シームレス接続は、ファイルシステムによってデータの

連続供給が保証されるように作られなければならない。これは、Clip1 (Clip AV ストリームファイル) と Clip2 (Clip AV ストリームファイル) に接続される Bridge-Clip AV ストリームを、データアロケーション規定を満たすように配置することによって行われなければならない。

## 【 0 2 8 8 】

RSPN\_exit\_from\_previous\_Clip 以前の Clip1 (Clip AV ストリームファイル) のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN\_exit\_from\_previous\_Clip が選択されなければならない。Bridge-Clip AV ストリームのデータ長は、ハーフフラグメント以上の連続領域に配置されるように、選択されなければならない。RSPN\_enter\_to\_current\_Clip 以後の Clip2 (Clip AV ストリームファイル) のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN\_enter\_to\_current\_Clip が選択されなければならない。

## 【 0 2 8 9 】

Bridge-Clip を使用しないでシームレス接続する場合のデータアロケーションの制限について、図 9 5 を参照して説明するに、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1 (Clip AV ストリームファイル) の最後の部分と Clip2 (Clip AV ストリームファイル) の最初の部分を、データアロケーション規定を満たすように配置することによって行われなければならない。

## 【 0 2 9 0 】

Clip1 (Clip AV ストリームファイル) の最後のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。Clip2 (Clip AV ストリームファイル) の最初のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。

## 【 0 2 9 1 】

次に、DVR-STD について説明する。DVR-STD は、DVR MPEG2 トランスポートストリームの生成および検証の際におけるデコード処理をモデル化するための概念モデルである。また、DVR-STD は、上述したシームレス接続された 2 つの PlayItem に

よって参照されるAVストリームの生成および検証の際におけるデコード処理をモデル化するための概念モデルでもある。

#### 【 0 2 9 2 】

DVR-STDモデルを図9 6に示す。図9 6に示したモデルには、DVR MPEG-2トランスポートストリームプレーヤモデルが構成要素として含まれている。 $n$ ,  $TB_n$ ,  $MB_n$ ,  $EB_n$ ,  $TB_{sys}$ ,  $B_{sys}$ ,  $R_{xn}$ ,  $R_{bxn}$ ,  $R_{xsys}$ ,  $D_n$ ,  $D_{sys}$ ,  $O_n$ および $P_n(k)$ の表記方法は、ISO/IEC13818-1のT-STDに定義されているものと同じである。すなわち、次の通りである。 $n$ は、エレメンタリーストリームのインデクス番号である。 $TB_n$ は、エレメンタリーストリーム $n$ のトランスポートバッファである。

#### 【 0 2 9 3 】

$MB_n$ は、エレメンタリーストリーム $n$ の多重バッファである。ビデオストリームについてのみ存在する。 $EB_n$ は、エレメンタリーストリーム $n$ のエレメンタリーストリームバッファである。ビデオストリームについてのみ存在する。 $TB_{sys}$ は、復号中のプログラムのシステム情報のための入力バッファである。 $B_{sys}$ は、復号中のプログラムのシステム情報のためのシステムターゲットデコーダ内のメインバッファである。 $R_{xn}$ は、データが $TB_n$ から取り除かれる伝送レートである。 $R_{bxn}$ は、PESパケットペイロードが $MB_n$ から取り除かれる伝送レートである。ビデオストリームについてのみ存在する。

#### 【 0 2 9 4 】

$R_{xsys}$ は、データが $TB_{sys}$ から取り除かれる伝送レートである。 $D_n$ は、エレメンタリーストリーム $n$ のデコーダである。 $D_{sys}$ は、復号中のプログラムのシステム情報に関するデコーダである。 $O_n$ は、ビデオストリーム $n$ のre-ordering bufferである。 $P_n(k)$ は、エレメンタリーストリーム $n$ の $k$ 番目のプレゼンテーションユニットである。

#### 【 0 2 9 5 】

DVR-STDのデコーディングプロセスについて説明する。単一のDVR MPEG-2トランスポートストリームを再生している間は、トランスポートパケットを $TB_1$ ,  $TB_n$ または $TB_{sys}$ のバッファへ入力するタイミングは、ソースパケットのarrival\_time\_stampにより決定される。 $TB_1$ ,  $MB_1$ ,  $EB_1$ ,  $TB_n$ ,  $B_n$ ,  $TB_{sys}$ および $TB_{sys}$ のバッ

ファリング動作の規定は、ISO/IEC 13818-1に規定されているT-STDと同じである。復号動作と表示動作の規定もまた、ISO/IEC 13818-1に規定されているT-STDと同じである。

#### 【0 2 9 6】

シームレス接続されたPlayItemを再生している間のデコーディングプロセスについて説明する。ここでは、シームレス接続されたPlayItemによって参照される2つのAVストリームの再生について説明をすることにし、以後の説明では、上述した（例えば、図88に示した）TS1とTS2の再生について説明する。TS1は、先行するストリームであり、TS2は、現在のストリームである。

#### 【0 2 9 7】

図97は、あるAVストリーム（TS1）からそれにシームレスに接続された次のAVストリーム（TS2）へと移る時のトランスポートパケットの入力、復号、表示のタイミングチャートを示す。所定のAVストリーム（TS1）からそれにシームレスに接続された次のAVストリーム（TS2）へと移る間には、TS2のアライバルタイムベースの時間軸（図97においてATC2で示される）は、TS1のアライバルタイムベースの時間軸（図97においてATC1で示される）と同じでない。

#### 【0 2 9 8】

また、TS2のシステムタイムベースの時間軸（図97においてSTC2で示される）は、TS1のシステムタイムベースの時間軸（図97においてSTC1で示される）と同じでない。ビデオの表示は、シームレスに連続していることが要求される。オーディオのプレゼンテーションユニットの表示時間にはオーバーラップがあっても良い。

#### 【0 2 9 9】

DVR-STD への入力タイミングについて説明する。時刻 $T_1$ までの時間、すなわち、TS1の最後のビデオパケットがDVR-STDのTB1に入力終了するまでは、DVR-STDのTB1、TBn またはTBsysのバッファへの入力タイミングは、TS1のソースパケットのarrival\_time\_stampによって決定される。

#### 【0.3 0 0】

TS1の残りのパケットは、TS\_recording\_rate(TS1)のビットレートでDVR-STDの

TBnまたはTBsysのバッファへ入力されなければならない。ここで、TS\_recording\_rate(TS1)は、Clip1に対応するClipInfo()において定義されるTS\_recording\_rateの値である。TS1の最後のバイトがバッファへ入力する時刻は、時刻 $T_2$ である。従って、時刻 $T_1$ から $T_2$ までの区間では、ソースパケットのarrival\_time\_stampは無視される。

## 【 0 3 0 1 】

N1をTS1の最後のビデオパケットに続くTS1のトランスポートパケットのバイト数とすると、時刻 $T_1$ 乃至 $T_2$ までの時間DT1は、N1バイトがTS\_recording\_rate(TS1)のビットレートで入力終了するために必要な時間であり、次式により算出される。

$$DT1 = T_2 - T_1 = N1 / TS\_recording\_rate(TS1)$$

時刻 $T_1$ 乃至 $T_2$ までの間は、RXnとRXsysの値は共に、TS\_recording\_rate(TS1)の値に変化する。このルール以外のバッファリング動作は、T-STDと同じである。

## 【 0 3 0 2 】

$T_2$ の時刻において、arrival time clock counterは、TS2の最初のソースパケットのarrival\_time\_stampの値にリセットされる。DVR-STDのTB1, TBn またはTBsysのバッファへの入力タイミングは、TS2のソースパケットのarrival\_time\_stampによって決定される。RXnとRXsysは共に、T-STDにおいて定義されている値に変化する。

## 【 0 3 0 3 】

付加的なオーディオバッファリングおよびシステムデータバッファリングについて説明するに、オーディオデコーダとシステムデコーダは、時刻 $T_1$ から $T_2$ までの区間の入力データを処理することができるよう、T-STDで定義されるバッファ量に加えて付加的なバッファ量（約1秒分のデータ量）が必要である。

## 【 0 3 0 4 】

ビデオのプレゼンテーションタイミングについて説明するに、ビデオプレゼンテーションユニットの表示は、接続点を通して、ギャップなしに連続でなければならない。ここで、STC1は、TS1のシステムタイムベースの時間軸（図9 7ではSTC1と図示されている）とし、STC2は、TS2のシステムタイムベースの時間軸（図

9 7 ではSTC2と図示されている。正確には、STC2は、TS2の最初のPCRがT-STDに  
入力した時刻から開始する。) とする。

【 0 3 0 5 】

STC1とSTC2の間のオフセットは、次のように決定される。PTS<sup>1</sup><sub>end</sub>は、TS1の最  
後のビデオプレゼンテーションユニットに対応するSTC1上のPTSであり、PTS<sup>2</sup><sub>sta  
rt</sub>は、TS2の最初のビデオプレゼンテーションユニットに対応するSTC2上のPTSで  
あり、T<sub>pp</sub>は、TS1の最後のビデオプレゼンテーションユニットの表示期間とする  
と、2つのシステムタイムベースの間のオフセットSTC\_deltaは、次式により算  
出される。

$$STC\_delta = PTS^1_{end} + T_{pp} - PTS^2_{start}$$

【 0 3 0 6 】

オーディオのプレゼンテーションのタイミングについて説明するに、接続点に  
おいて、オーディオプレゼンテーションユニットの表示タイミングのオーバーラ  
ップがあっても良く、それは0乃至2オーディオフレーム未満である（図9 7に  
図示されている”audio overlap”を参照）。どちらのオーディオサンプルを選択  
するかということと、オーディオプレゼンテーションユニットの表示を接続点の  
後の補正されたタイムベースに再同期することは、プレーヤ側により設定される  
ことである。

【 0 3 0 7 】

DVR-STDのシステムタイムクロックについて説明するに、時刻T<sub>5</sub>において、TS  
1の最後のオーディオプレゼンテーションユニットが表示される。システムタイ  
ムクロックは、時刻T<sub>2</sub>からT<sub>5</sub>の間にオーバーラップしていても良い。この区間  
では、DVR-STDは、システムタイムクロックを古いタイムベースの値（STC1）と  
新しいタイムベースの値（STC2）の間で切り替える。STC2の値は、次式により算  
出される。

$$STC2 = STC1 - STC\_delta$$

【 0 3 0 8 】

バッファリングの連続性について説明する。STC1<sup>1</sup><sub>video\_end</sub>は、TS1の最後の  
ビデオパケットの最後のバイトがDVR-STDのTB1へ到着する時のシステムタイムベ

ースSTC1上のSTCの値である。STC2<sup>2</sup><sub>video\_start</sub>は、TS2の最初のビデオパケットの最初のバイトがDVR-STDのTB1へ到着する時のシステムタイムベースSTC2上のSTCの値である。STC2<sup>1</sup><sub>video\_end</sub>は、STC1<sup>1</sup><sub>video\_end</sub>の値をシステムタイムベースSTC2上の値に換算した値である。STC2<sup>1</sup><sub>video\_end</sub>は、次式により算出される。

$$STC2^1_{video\_end} = STC1^1_{video\_end} - STC\_delta$$

【0 3 0 9】

DVR-STDに従うために、次の2つの条件を満たす事が要求される。まず、TS2の最初のビデオパケットのTB1への到着タイミングは、次に示す不等式を満たさなければならない。そして、次に示す不等式を満たさなければならない。

$$STC2^2_{video\_start} > STC2^1_{video\_end} + \Delta T_1$$

この不等式が満たされるように、Clip1および、または、Clip2の部分的なストリームを再エンコードおよび、または、再多重化する必要がある場合は、その必要に応じて行われる。

【0 3 1 0】

次に、STC1とSTC2を同じ時間軸上に換算したシステムタイムベースの時間軸上において、TS1からのビデオパケットの入力とそれに続くTS2からのビデオパケットの入力は、ビデオバッファをオーバーフローおよびアンダーフローさせてはならない。

【0 3 1 1】

このようなシンタクス、データ構造、規則に基づく事により、記録媒体に記録されているデータの内容、再生情報などを適切に管理することができ、もって、ユーザが再生時に適切に記録媒体に記録されているデータの内容を確認したり、所望のデータを簡便に再生できるようにすることができる。

【0 3 1 2】

上述した一連の処理は、ハードウェアにより実行させることもできるが、ソフトウェアにより実行させることもできる。一連の処理をソフトウェアにより実行させる場合には、そのソフトウェアを構成するプログラムが専用のハードウェアに組み込まれているコンピュータ、または、各種のプログラムをインストールすることで、各種の機能を実行することが可能な、例えば汎用のパーソナルコンピ

ュータなどに、記録媒体からインストールされる。

【 0 3 1 3 】

この記録媒体は、図 9 8 に示すように、コンピュータとは別に、ユーザにプログラムを提供するために配布される、プログラムが記録されている磁気ディスク 2 2 1 (フロッピディスクを含む)、光ディスク 2 2 2 (CD-ROM (Compact Disk -Read Only Memory) , DVD (Digital Versatile Disk) を含む)、光磁気ディスク 2 2 3 (MD (Mini-Disk) を含む)、若しくは半導体メモリ 2 2 4 などよりなるパッケージメディアにより構成されるだけでなく、コンピュータに予め組み込まれた状態でユーザに提供される、プログラムが記憶されている ROM 2 0 2 や記憶部 2 0 8 が含まれるハードディスクなどで構成される。

【 0 3 1 4 】

なお、本明細書において、媒体により提供されるプログラムを記述するステップは、記載された順序に従って、時系列的に行われる処理は勿論、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理をも含むものである。

【 0 3 1 5 】

また、本明細書において、システムとは、複数の装置により構成される装置全体を表すものである。

【 0 3 1 6 】

【発明の効果】

以上の如く、請求項 1 に記載の情報処理装置、請求項 4 に記載の情報処理方法、および請求項 5 に記載の記録媒体によれば、第 1 の AV ストリームから第 2 の AV ストリームへ連続的に再生されるように指示された場合、第 1 の AV ストリームの所定の部分と第 2 の AV ストリームの所定の部分から構成され、第 1 の AV ストリームから第 2 の AV ストリームに再生が切り換えられるとき再生される第 3 の AV ストリームを生成し、第 1 の生成手段により生成された第 3 の AV ストリームの情報として、第 1 の AV ストリームから第 3 の AV ストリームに再生が切り替わるとき再生される第 1 の AV ストリームのソースパケットのアドレスの情報と、第 3 の AV ストリームから第 2 の AV ストリームに再生が切り替わるとき再生される第 2 の AV スト



リームのソースパケットのアドレスの情報から構成されるアドレス情報を生成するようにしたので、別々に記録されたAVストリームの連続性を保つように再生できる。

【図面の簡単な説明】

【図 1】

本発明を適用した記録再生装置の一実施の形態の構成を示す図である。

【図 2】

記録再生装置 1 により記録媒体に記録されるデータのフォーマットについて説明する図である。

【図 3】

Real PlayListとVirtual PlayListについて説明する図である。

【図 4】

Real PlayListの作成について説明する図である。

【図 5】

Real PlayListの削除について説明する図である。

【図 6】

アセンブル編集について説明する図である。

【図 7】

Virtual PlayListにサブパスを設ける場合について説明する図である。

【図 8】

PlayListの再生順序の変更について説明する図である。

【図 9】

PlayList上のマークとClip上のマークについて説明する図である。

【図 1 0】

メニューサムネイルについて説明する図である。

【図 1 1】

PlayListに付加されるマークについて説明する図である。

【図 1 2】

クリップに付加されるマークについて説明する図である。

【図 1 3】

PlayList、Clip、サムネイルファイルの関係について説明する図である。

【図 1 4】

ディレクトリ構造について説明する図である。

【図 1 5】

info.dvrのシンタクスを示す図である。

【図 1 6】

DVR volumeのシンタクスを示す図である。

【図 1 7】

Resumevolumeのシンタクスを示す図である。

【図 1 8】

UIAppInfovolumeのシンタクスを示す図である。

【図 1 9】

Character set valueのテーブルを示す図である。

【図 2 0】

TableOfPlayListのシンタクスを示す図である。

【図 2 1】

TableOfPlayListの他のシンタクスを示す図である。

【図 2 2】

MakersPrivateDataのシンタクスを示す図である。

【図 2 3】

xxxxx.rplsとyyyyy.vplsのシンタクスを示す図である。

【図 2 4】

PlayListについて説明する図である。

【図 2 5】

PlayListのシンタクスを示す図である。

【図 2 6】

PlayList\_typeのテーブルを示す図である。

【図 2 7】

UIAppinfoPlayListのシンタクスを示す図である。

【図 2 8】

図 2 7 に示したUIAppinfoPlayListのシンタクス内のフラグについて説明する図である。

【図 2 9】

PlayItemについて説明する図である。

【図 3 0】

PlayItemについて説明する図である。

【図 3 1】

PlayItemについて説明する図である。

【図 3 2】

PlayItemのシンタクスを示す図である。

【図 3 3】

IN\_timeについて説明する図である。

【図 3 4】

OUT\_timeについて説明する図である。

【図 3 5】

Connection\_Conditionのテーブルを示す図である。

【図 3 6】

Connection\_Conditionについて説明する図である。

【図 3 7】

BridgeSequenceInfoを説明する図である。

【図 3 8】

BridgeSequenceInfoのシンタクスを示す図である。

【図 3 9】

SubPlayItemについて説明する図である。

【図 4 0】

SubPlayItemのシンタクスを示す図である。

【図 4 1】

SubPath\_typeのテーブルを示す図である。

【図 4 2】

PlayListMarkのシンタクスを示す図である。

【図 4 3】

Mark\_typeのテーブルを示す図である。

【図 4 4】

Mark\_time\_stampを説明する図である。

【図 4 5】

zzzzz.clipのシンタクスを示す図である。

【図 4 6】

ClipInfoのシンタクスを示す図である。

【図 4 7】

Clip\_stream\_typeのテーブルを示す図である。

【図 4 8】

offset\_SPNについて説明する図である。

【図 4 9】

offset\_SPNについて説明する図である。

【図 5 0】

STC 区間について説明する図である。

【図 5 1】

STC\_Infoについて説明する図である。

【図 5 2】

STC\_Infoのシンタクスを示す図である。

【図 5 3】

ProgramInfoを説明する図である。

【図 5 4】

ProgramInfoのシンタクスを示す図である。

【図 5 5】

VideoCondngInfoのシンタクスを示す図である。

【図 5 6】

Video\_formatのテーブルを示す図である。

【図 5 7】

frame\_rateのテーブルを示す図である。

【図 5 8】

display\_aspect\_ratioのテーブルを示す図である。

【図 5 9】

AudioCondngInfoのシンタクスを示す図である。

【図 6 0】

audio\_codingのテーブルを示す図である。

【図 6 1】

audio\_component\_typeのテーブルを示す図である。

【図 6 2】

sampling\_frequencyのテーブルを示す図である。

【図 6 3】

CPIについて説明する図である。

【図 6 4】

CPIについて説明する図である。

【図 6 5】

CPIのシンタクスを示す図である。

【図 6 6】

CPI\_typeのテーブルを示す図である。

【図 6 7】

ビデオEP\_mapについて説明する図である。

【図 6 8】

EP\_mapについて説明する図である。

【図 6 9】

EP\_mapについて説明する図である。

【図 7 0】

EP\_mapのシンタクスを示す図である。

【図 7 1】

EP\_type valuesのテーブルを示す図である。

【図 7 2】

EP\_map\_for\_one\_stream\_PIDのシンタクスを示す図である。

【図 7 3】

TU\_mapについて説明する図である。

【図 7 4】

TU\_mapのシンタクスを示す図である。

【図 7 5】

ClipMarkのシンタクスを示す図である。

【図 7 6】

mark\_typeのテーブルを示す図である。

【図 7 7】

mark\_type\_stampのテーブルを示す図である。

【図 7 8】

menu.thmbとmark.thmbのシンタクスを示す図である。

【図 7 9】

Thumbnailのシンタクスを示す図である。

【図 8 0】

thumbnail\_picture\_formatのテーブルを示す図である。

【図 8 1】

tn\_blockについて説明する図である。

【図 8 2】

DVR MPEG 2 のトランスポートストリームの構造について説明する図である。

【図 8 3】

DVR MPEG 2 のトランスポートストリームのレコーダモデルを示す図である。

【図 8 4】

DVR MPEG 2 のトランスポートストリームのプレーヤモデルを示す図である。

【図 8 5】

source packetのシンタクスを示す図である。

【図 8 6】

TP\_extra\_headerのシンタクスを示す図である。

【図 8 7】

copy permission indicatorのテーブルを示す図である。

【図 8 8】

シームレス接続について説明する図である。

【図 8 9】

シームレス接続について説明する図である。

【図 9 0】

シームレス接続について説明する図である

【図 9 1】

シームレス接続について説明する図である。

【図 9 2】

シームレス接続について説明する図である

【図 9 3】

オーディオのオーバーラップについて説明する図である。

【図 9 4】

BridgeSequenceを用いたシームレス接続について説明する図である。

【図 9 5】

BridgeSequenceを用いないシームレス接続について説明する図である。

【図 9 6】

DVR STDモデルを示す図である。

【図 9 7】

復号、表示のタイミングチャートを示す図である。

【図 9 8】

媒体を説明する図である。

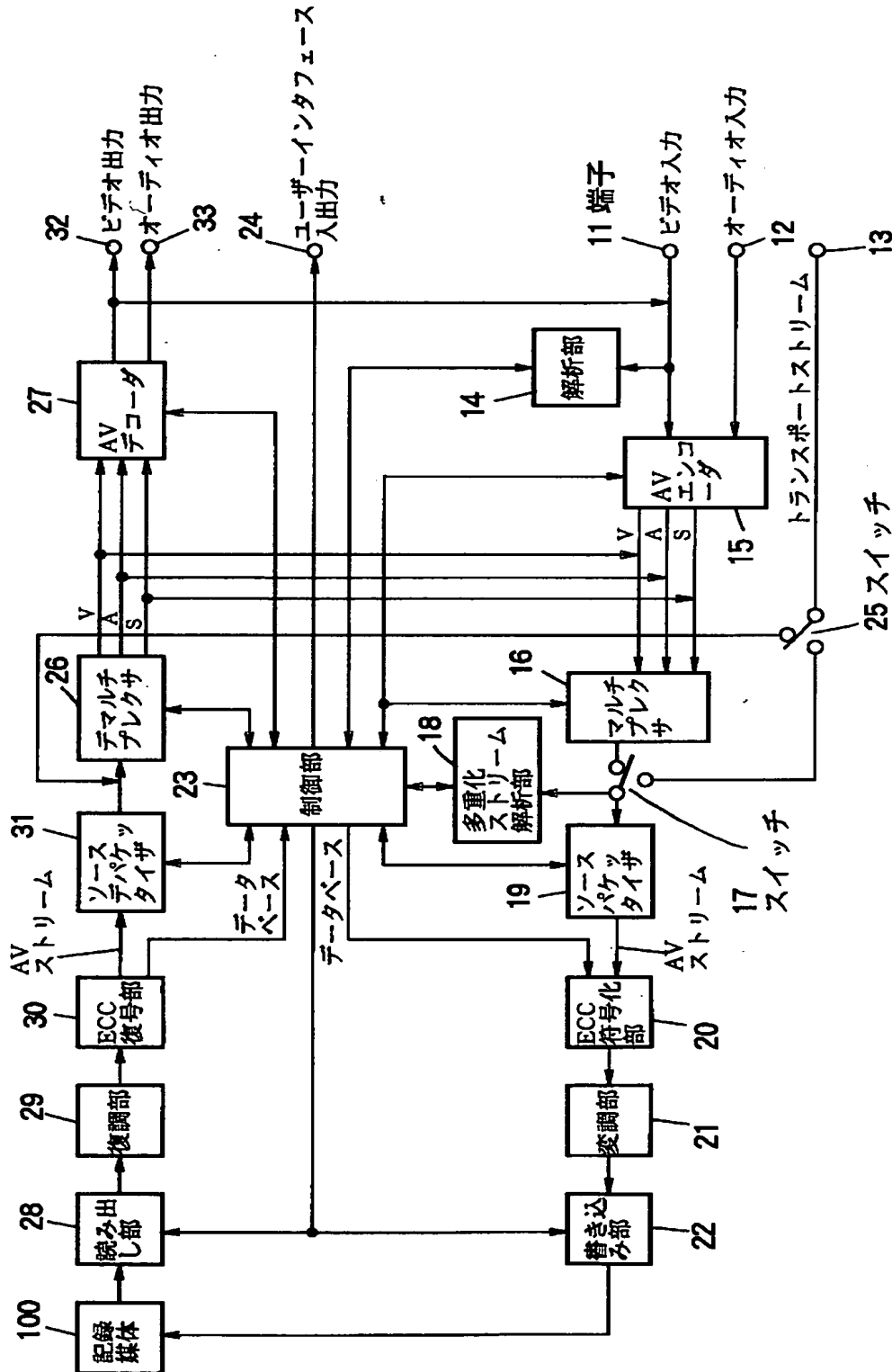
【符号の説明】

1 記録再生装置, 11乃至13 端子, 14 解析部, 15 AVエン  
コーダ, 16 マルチプレクサ, 17 スイッチ, 18 多重化ストリー  
ム解析部, 19 ソースパケッタイザ, 20 ECC符号化部, 21 変調  
部, 22 書き込み部, 23 制御部, 24 ユーザインタフェース,  
25 スイッチ, 26 デマルチプレクサ, 27 AVデコーダ, 28 読  
み出し部, 29 復調部, 30 ECC復号部, 31 ソースパケッタイザ  
, 32, 33 端子



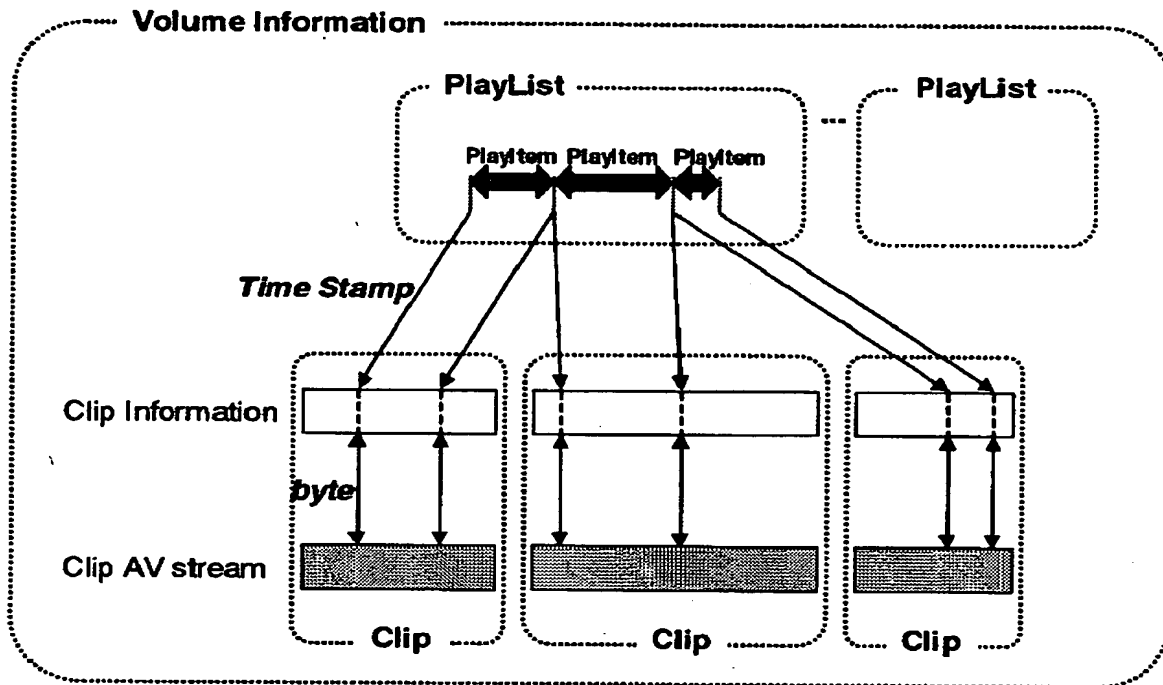
【書類名】 図面

【図 1】

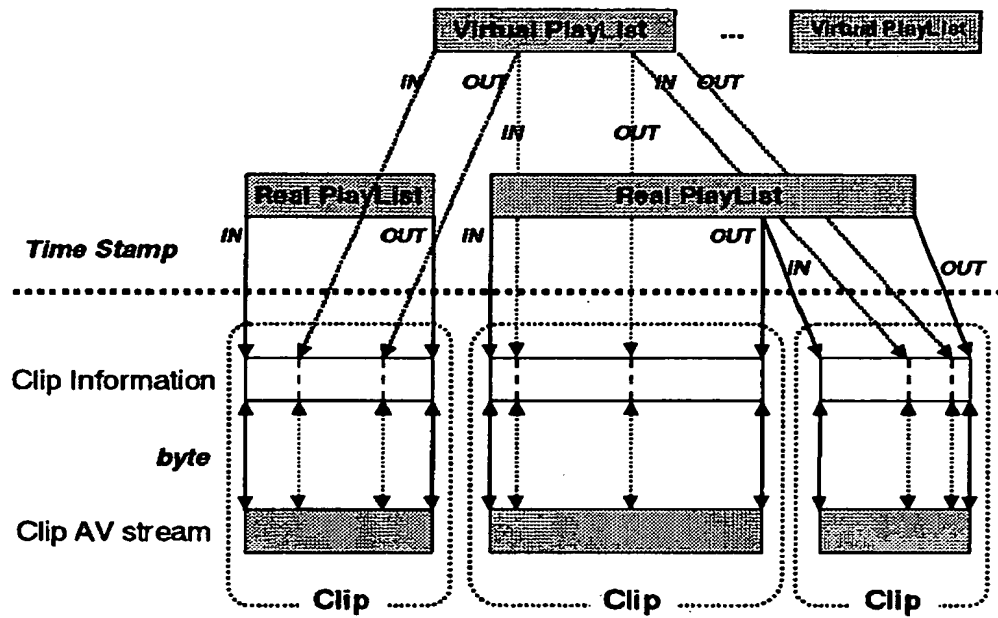


記録再生装置 1

【図 2】

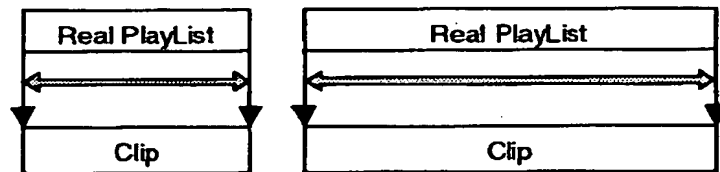


【図 3】



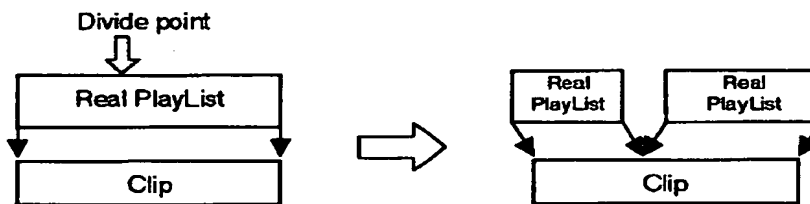
【図 4】

(A)



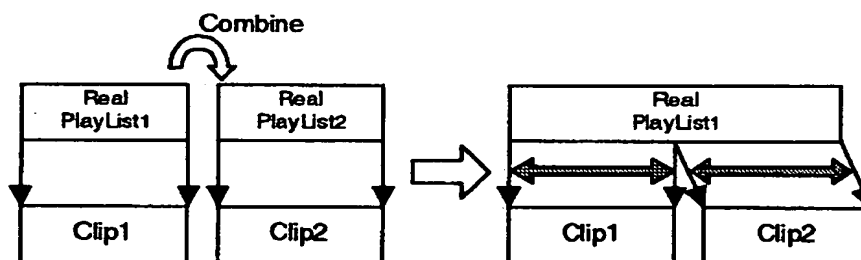
Real PlayList のクリエイトの例

(B)



Real PlayList のディバイドの例

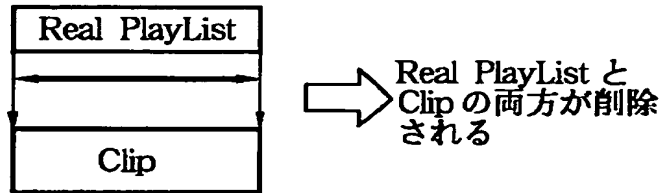
(C)



Real PlayList のコンバインの例

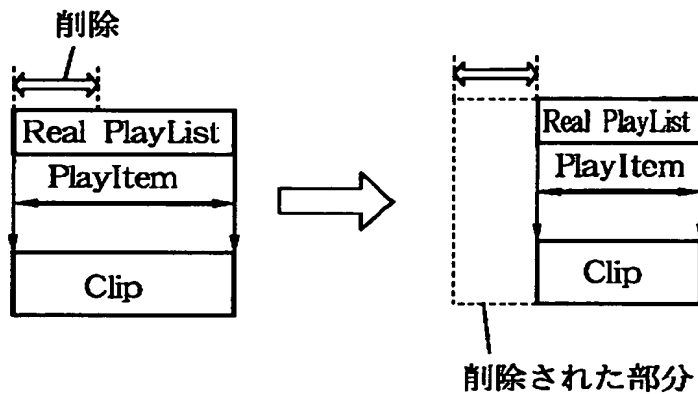
【図 5】

(A)



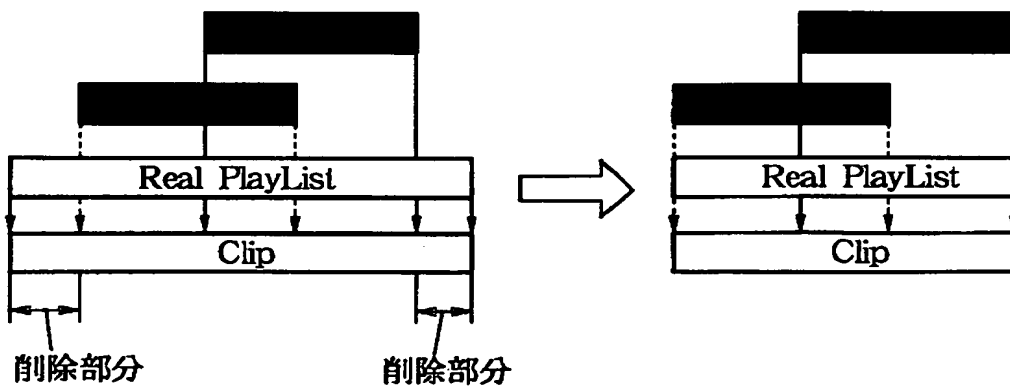
Real Playlist 全体のデリートの例

(B)



Real Playlist の部分的なデリートの例

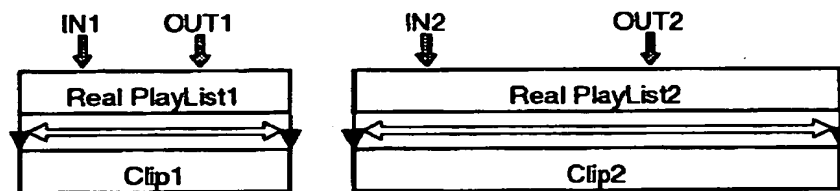
(C)



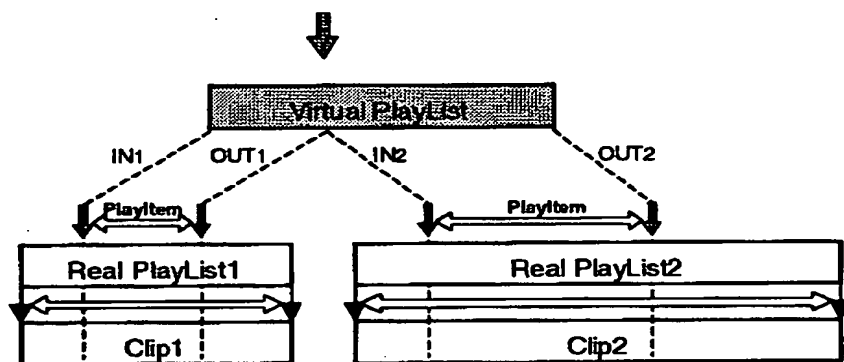
Real Playlist のミニマイズの例

【図 6】

(A)

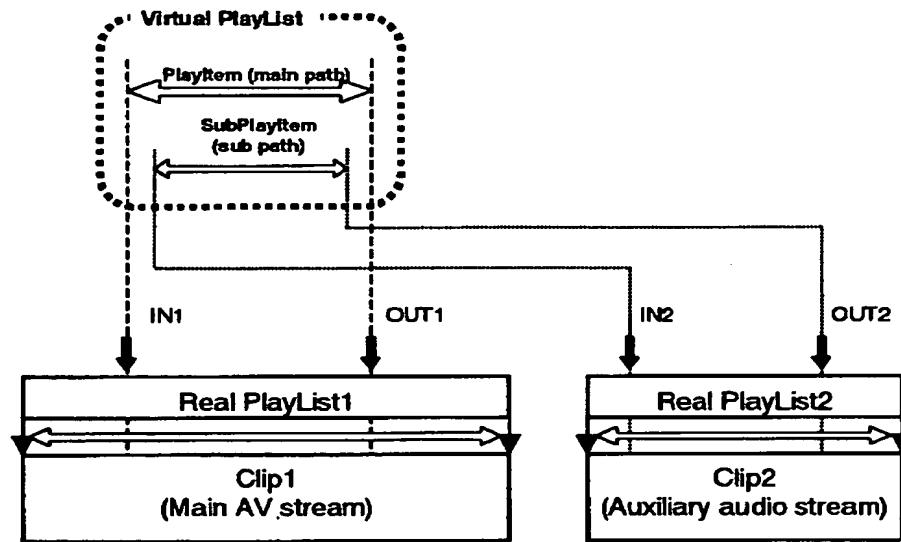


(B)



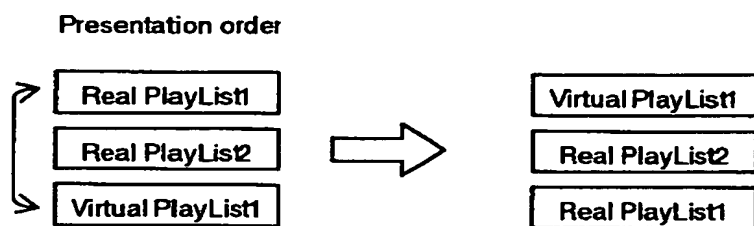
アセンブル編集の例

【図 7】



Virtual Playlist へのオーディオのアフレコの例

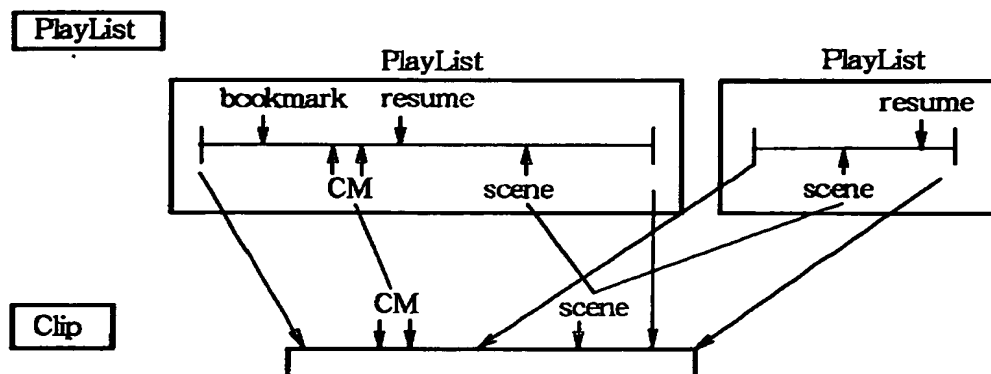
【図 8】



PlayList の再生順序の変更の例

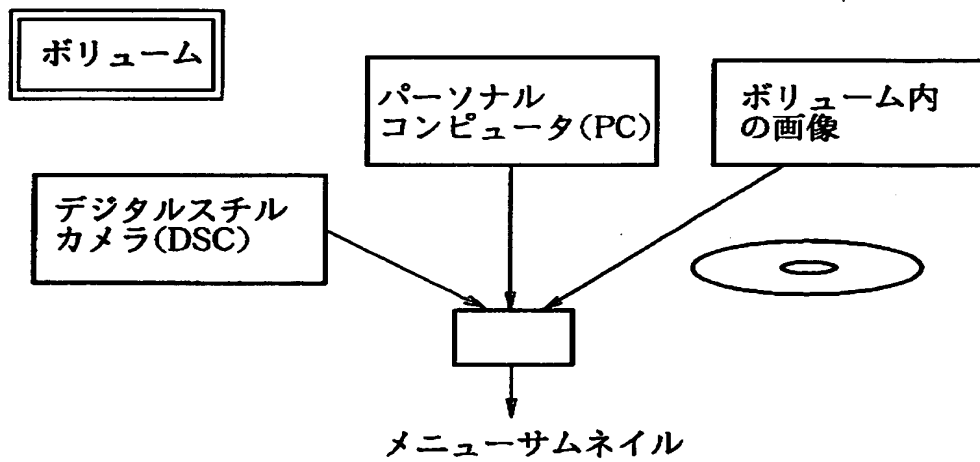


【図9】

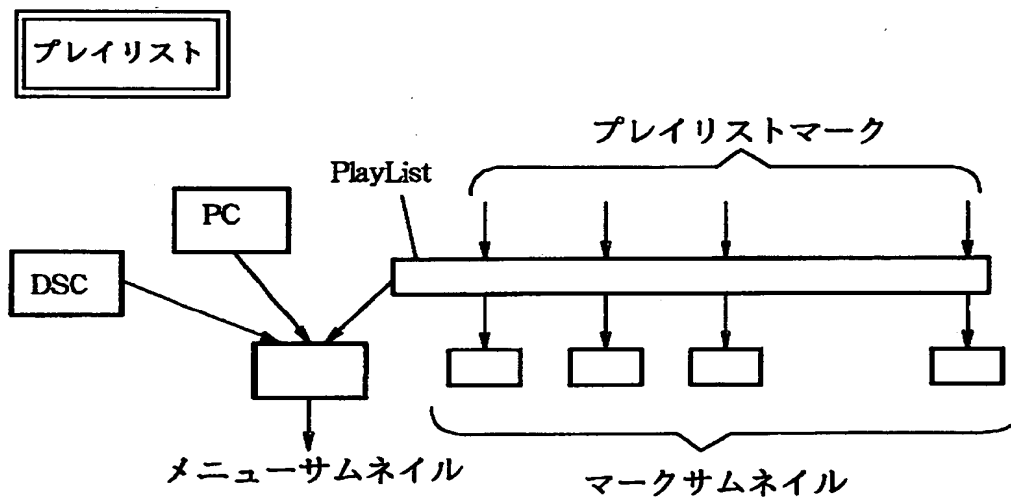


Playlist 上のマークと Clip 上のマーク

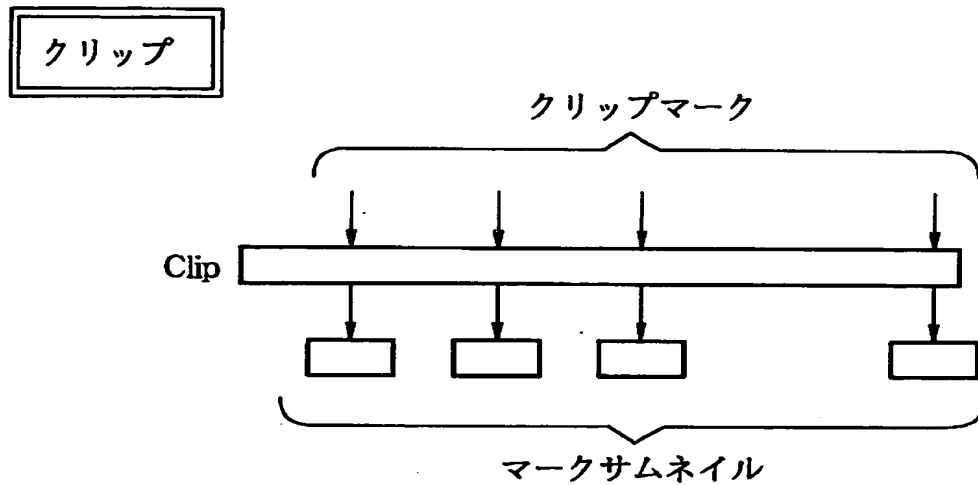
【図10】



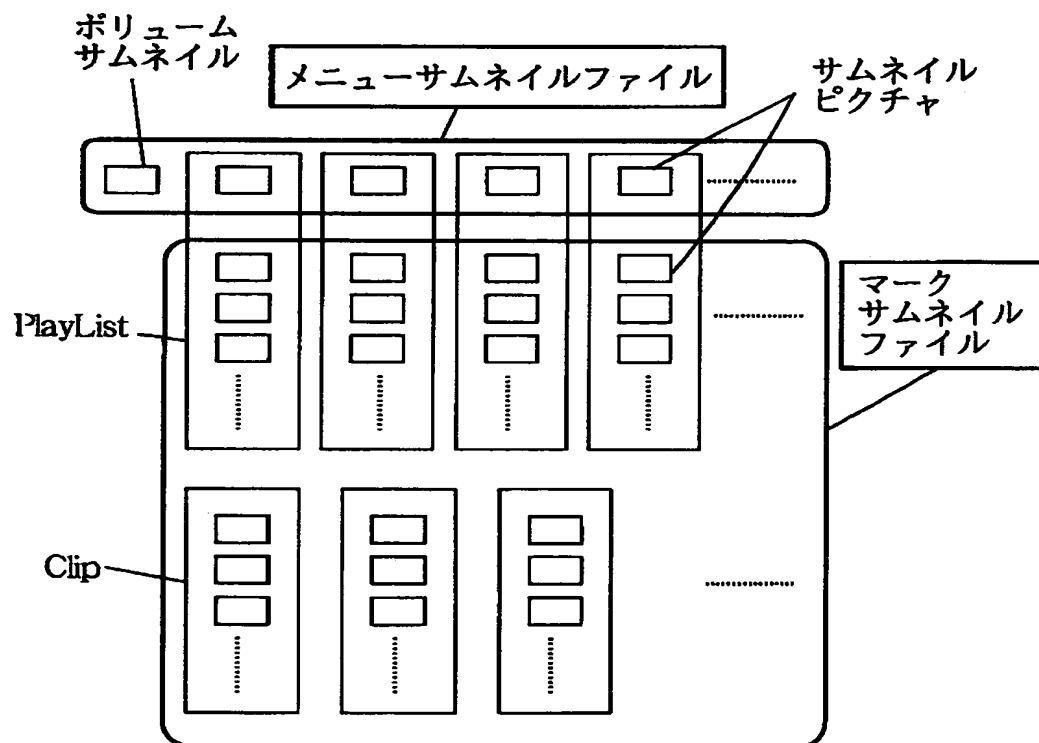
【図11】



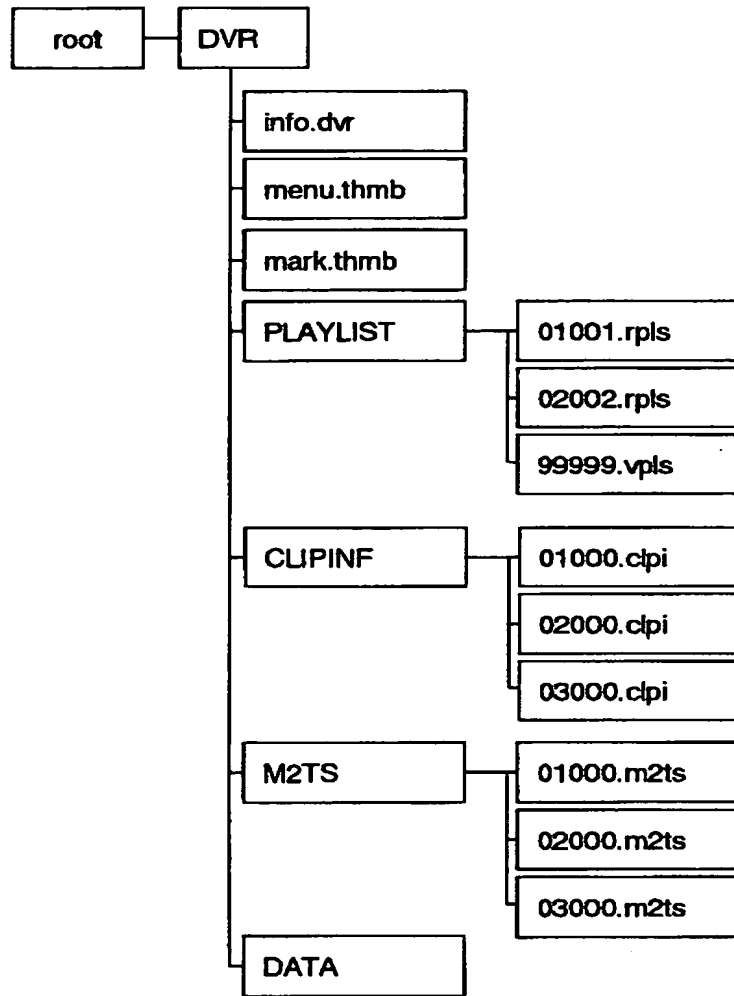
【図 12】



【図 13】



【図 1 4】



【図 1 5】

Syntax	No. of bits	Mnemonics
info.dvr {		
TableOfPlayLists Start_address	32	uimsbf
MakerPrivateData Start_address	32	uimsbf
reserved	192	bslbf
DVRVolume()		
for(i=0; i<N1; i++){		
padding_word	16	bslbf
}		
TableOfPlayLists()		
for(i=0; i<N2; i++){		
padding_word	16	bslbf
}		
MakerPrivateData()		
}		

info.dvr のシンタックス

【図 16】

Syntax	No. of bits	Mnemonics
DVRVolume() {		
version_number	8*4	bslbf
length	32	ulmsbf
ResumeVolume()		
UIAppInfoVolume()		
}		

DVR Volume のシンタックス

【図 1 7】

Syntax	No. of bits	Mnemonics
ResumeVolume() {		
reserved	15	bslbf
valid_flag	1	bslbf
resume_PlayList_name	8*10	bslbf
}		

ResumeVolume のシンタックス

【図 1 8】

Syntax	No. of bits	Mnemonics
UIAppInfoVolume {		
character_set	8	bslbf
name_length	8	uimbsbf
Volume_name	8*256	bslbf
reserved	15	bslbf
Volume_protect_flag	1	bslbf
PIN	8*4	bslbf
ref_thumbnail_index	16	uimbsbf
reserved_for_future_use	256	bslbf
}		

UIAppInfoVolume のシンタックス



【図 1 9】

Value	Character coding
0x00	Reserved
0x01	ISO/IEC 646 (ASCII)
0x02	ISO/IEC 10646-1 (Unicode)
0x03-0xff	Reserved

Character set value

【図 2 0】

Syntax	No. of bits	Mnemonics
TableOfPlayLists() {		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayLists	16	uimsbf
for (l=0; l<number_of_PlayLists; l++) {		
PlayList_file_name	8*10	bslbf
}		
}		

TableOfPlayLists のシンタックス

【図 2 1】

■ TableOfPlayLists - シンタックス (4.2.3.2 の別案)

Syntax	No. of bits	Mnemonics
TableOfPlayLists() {		
version_number	8*4	bs bf
length	32	ulmsbf
number_of_PlayLists	16	ulmsbf
for (i=0; i<number_of_PlayLists; i++) {		
PlayList_file_name	8*10	bs bf
UIAppInfoPlayList()		
}		
}		

TableOfPlayLists の別シンタックス

【図 2 2】

Syntax	No. of bits	Mnemonics
MakersPrivateData() {		
version_number	8*4	bslbf
length	32	uimsbf
if(length != 0){		
mpd_blocks_start_address	32	uimsbf
number_of_maker_entries	16	uimsbf
mpd_block_size	16	uimsbf
number_of_mpd_blocks	16	uimsbf
reserved	16	bslbf
for (i=0; i<number_of_maker_entries; i++){		
maker_ID	16	uimsbf
maker_model_code	16	uimsbf
start_mpd_block_number	16	uimsbf
reserved	16	bslbf
mpd_length	32	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for (j=0; j<number_of_mpd_blocks; j++){		
mpd_block	mpd_block_size*1024*8	
}		
}		
}		

MakersPrivateData のシンタックス

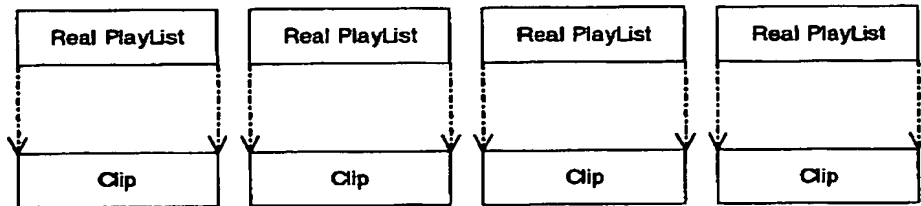
【図 2 3】

Syntax	No. of bits	Mnemonics
xxxx.rpls/yyyy.vpls {		
PlayListMark Start_address	32	uimbsbf
MakerPrivateData Start_address	32	uimbsbf
reserved	192	bslbf
PlayList()		
for( i=0; i<N1; i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for( i=0; i<N2; i++){		
padding_word	16	bslbf
}		
MakerPrivateData()		
}		

xxxxx.rpls と yyyyy.vpls のシンタクス

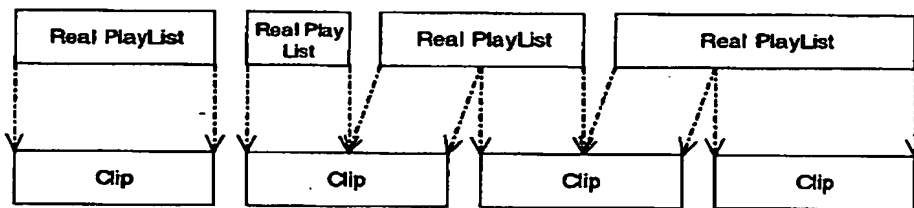
【図 2 4】

(A)



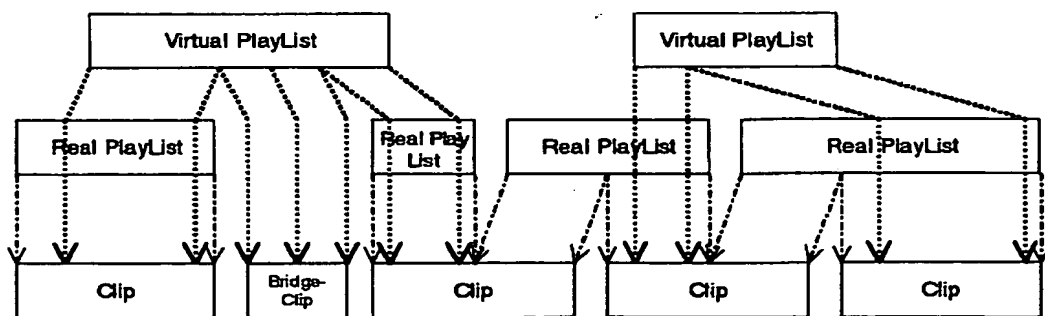
初めて AV ストリームが Clip として記録された時の Real PlayList の例

(B)



編集後の Real PlayList の例

(C)



Virtual PlayList の例

【図 2 5】

Syntax	No. of bits	Mnemonics
Playlist() {		
version_number	8*4	bslbf
length	32	uimsbf
Playlist_type	8	uimsbf
CPI_type	1	bslbf
reserved	7	bslbf
UIAppInfoPlaylist()		
number_of_PlayItems      // main path	16	uimsbf
if (<Virtual Playlist>) {		
number_of_SubPlayItems      // sub path	16	uimsbf
}else{		
reserved	16	bslbf
}		
for (PlayItem_id=0;		
PlayItem_id<number_of_PlayItems;		
PlayItem_id++) {		
PlayItem()		
// main path		
}		
if (<Virtual Playlist>) {		
if (CPI_type==0 && Playlist_type==0) {		
for (i = 0; i < number_of_SubPlayItems; i++)		
SubPlayItem()		
// sub path		
}		
}		
}		

Playlist のシンタックス

【図 2 6】

PlayList_type	Meaning
0	<b>AV 記録のための PlayList</b> この PlayList に参照されるすべての Clip は、一つ以上のビデオストリームを含まなければならない。
1	<b>オーディオ記録のための PlayList</b> この PlayList に参照されるすべての Clip は、一つ以上のオーディオストリームを含まなければならない、そしてビデオストリームを含んではならない。
2 - 255	reserved

PlayList\_type



【図 2 7】

Syntax	No. of bits	Mnemonics
UIAppInfoPlayList2() {		
character_set	8	bslbf
name_length	8	uimbsf
PlayList_name	8*256	bslbf
reserved	8	bslbf
record_time_and_date	4*14	bslbf
reserved	8	bslbf
duration	4*6	bslbf
valid_period	4*8	bslbf
maker_id	16	uimbsf
maker_code	16	uimbsf
reserved	11	bslbf
playback_control_flag	1	bslbf
write_protect_flag	1	bslbf
is_played_flag	1	bslbf
archive	2	bslbf
ref_thumbnail_index	16	uimbsf
reserved_for_future_use	256	bslbf
}		

UIAppInfoPlayList のシンタックス

【図 2 8】

(A)

write_protect_flag	Meaning
0b	その PlayList を自由に消去しても良い。
1b	write_protect_flag を除いてその PlayList の内容は、消去および変更されるべきではない。

write\_protect\_flag

(B)

is_played_flag	Meaning
0b	その PlayList は、記録されてから一度も再生されたことがない。
1b	PlayList は、記録されてから一度は再生された。

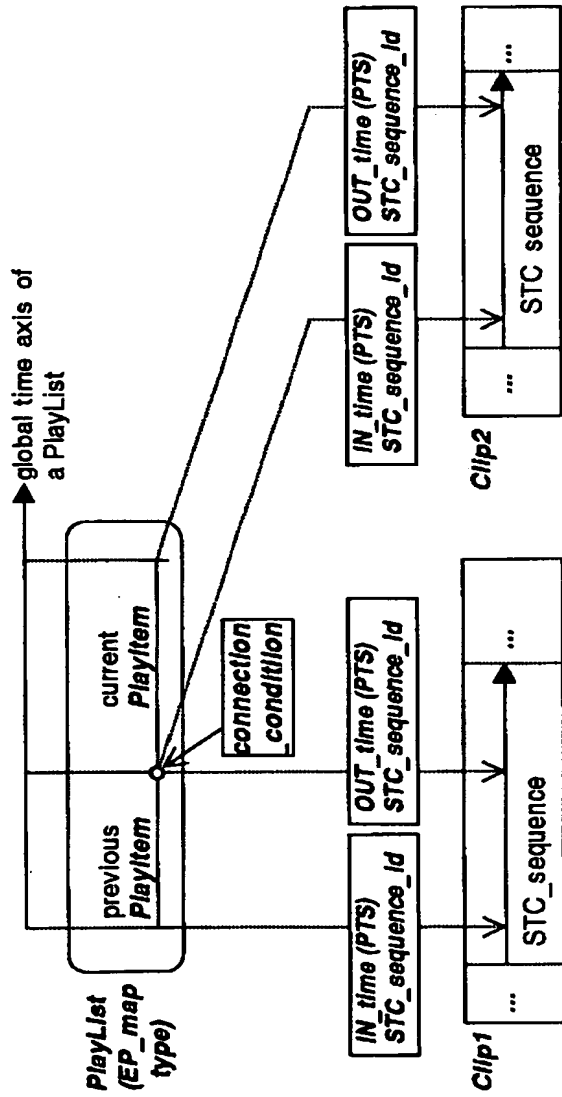
is\_played\_flag

(C)

archive	Meaning
00b	何も情報が定義されていない。
01b	オリジナル
10b	コピー
11b	reserved

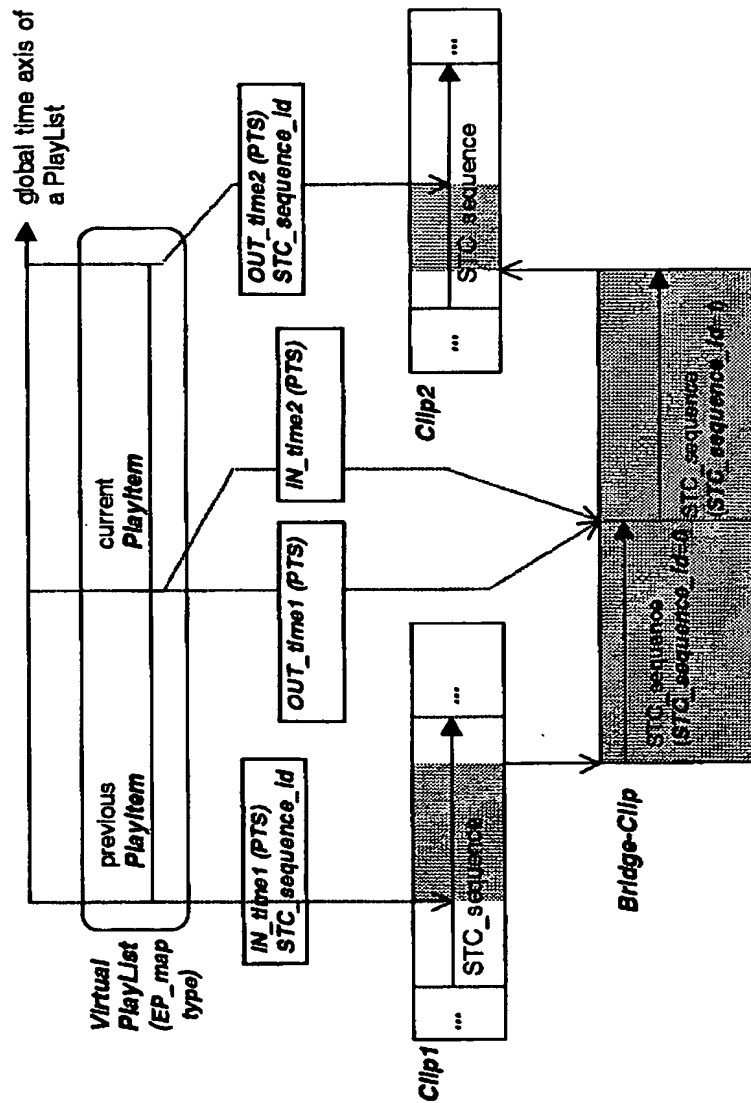
archive

【図 29】



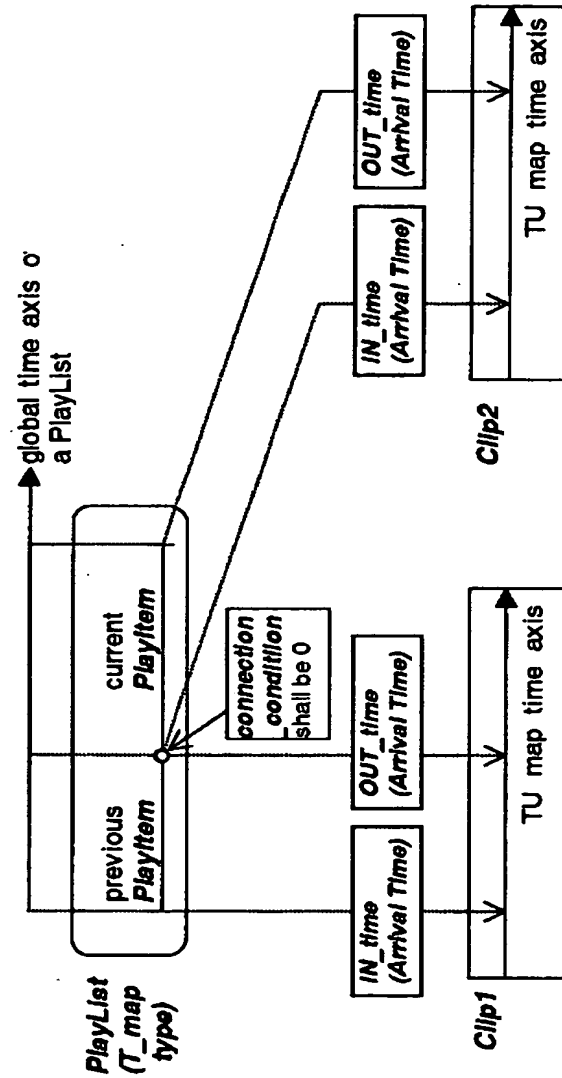
Playlist が EP\_map type であり、かつ PlayItem が BridgeSequence を持たない時の例

【図 30】



Playlist が EP\_map type であり、かつ PlayItem が BridgeSequence を持つ時の例

【図 31】



Playlist が TU\_map type である時の例

【図 3 2】

Syntax	No. of bits	Mnemonics
PlayItem() {		
Clip_Information_file_name	8*10	bslbf
reserved	24	bslbf
STC_sequence_id	8	ulmsbf
IN_time	32	ulmsbf
OUT_time	32	ulmsbf
reserved	14	bslbf
connection_condition	2	bslbf
if (<Virtual PlayList>) {		
if (connection_condition=='10') {		
BridgeSequenceInfo()		
}		
}		
}		

PlayItem のシンタックス

【図 3 3】

CPI_type in the PlayList()	Semantics of IN_time
EP_map type	IN_time は、PlayItem の中で最初のプレゼンテーションユニットに対応する 33 ビット長の PTS の上位 32 ビットを示さなければならない。
TU_map type	IN_time は、TU_map_time_axis 上の時刻でなければならない。かつ、IN_time は、time_unit の精度に丸めて表さなければならない。IN_time は、次に示す等式により計算される。  $IN\_time = TU\_start\_time \% 2^{32}$

IN\_time

【図 3 4】

CPI_type in the PlayList()	Semantics of OUT_time
EP_map type	<p>OUT_time は、次に示す等式によって計算される。Presentation_end_TS の値の上位 32 ビットを示さなければならない。</p> $Presentation\_end\_TS = PTS\_out + AU\_duration$ <p>ここで、</p> <p>PTS_out は、PlayItem の中で最後のプレゼンテーションユニットに対応する 33 ビット長の PTS である。</p> <p>AU_duration は、最後のプレゼンテーションユニットの 90kHz 単位の表示期間である。</p>
TU_map type	<p>OUT_time は、TU_map_time_axis 上の時刻でなければならない。かつ、OUT_time は、time_unit の精度に丸めて表さなければならない。</p> <p>OUT_time は、次に示す等式により計算される。</p> $OUT\_time = TU\_start\_time \% 2^{32}$

OUT\_time

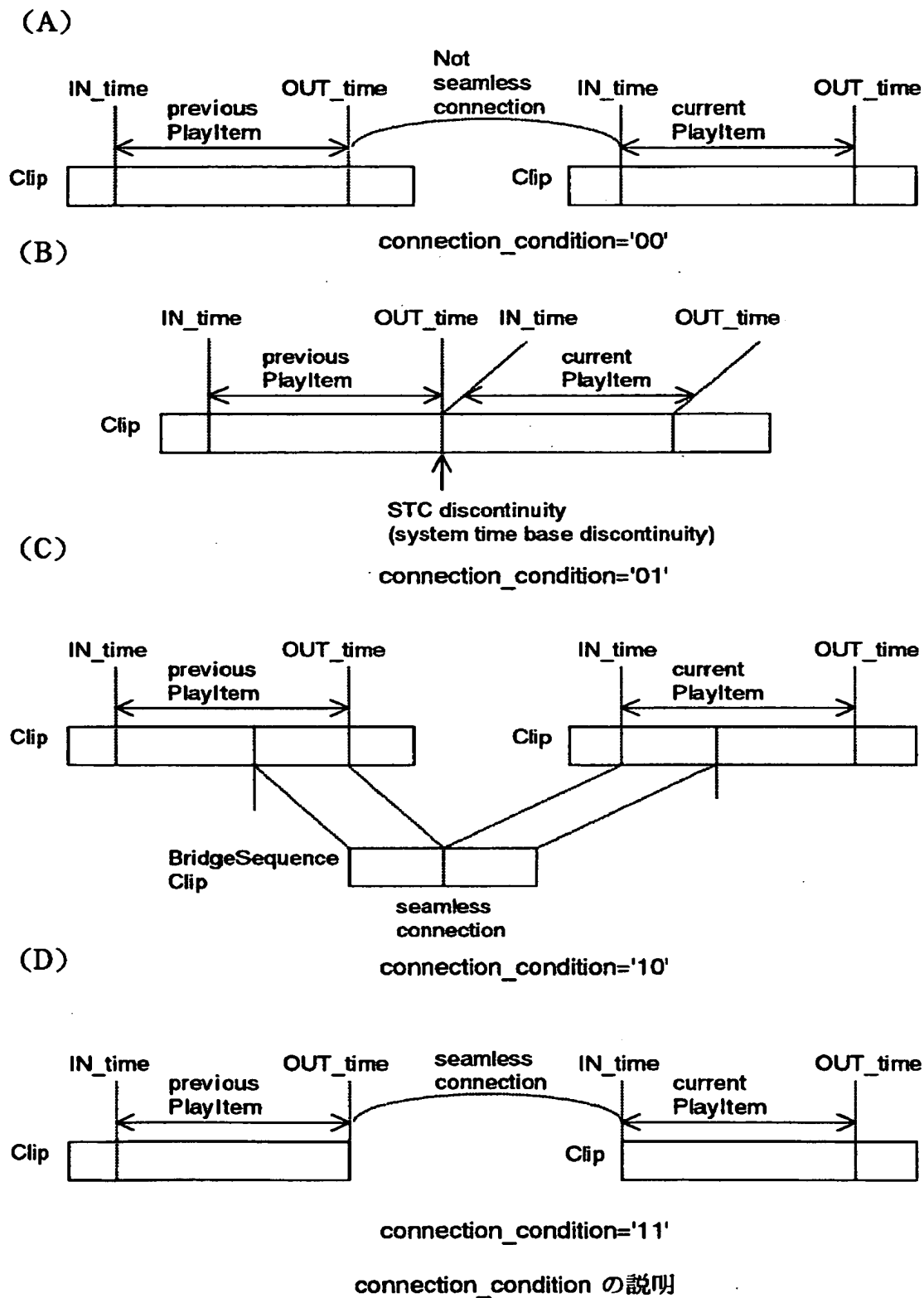


【図 3 5】

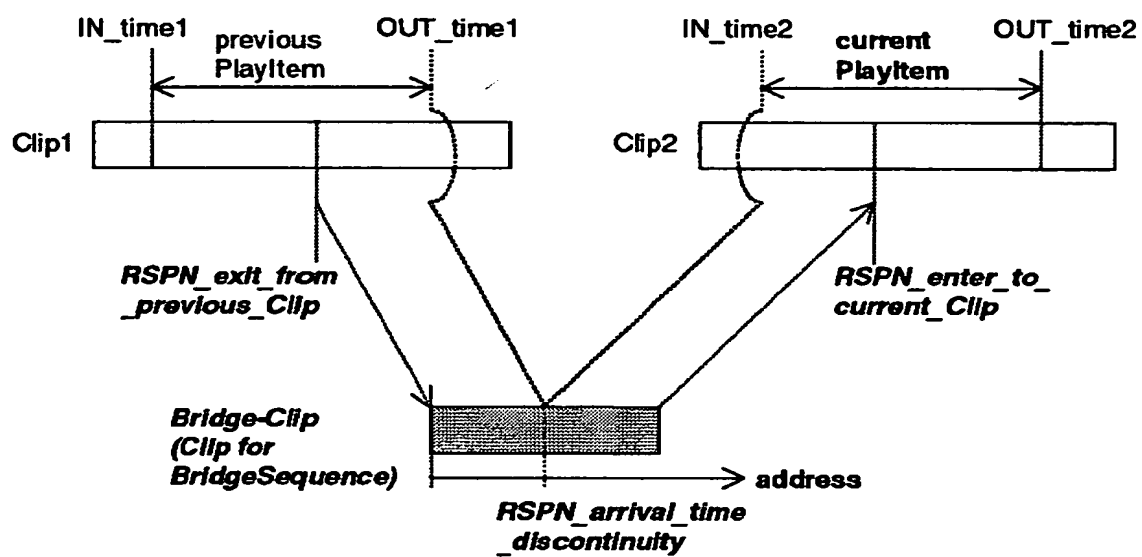
connection condition	meaning
00	<ul style="list-style-type: none"> <li>先行する PlayItem と現在の PlayItem の接続は、シームレス再生の保証がなされていない。</li> <li>PlayList の CPI_type が TU_map type である場合、connection_condition は、この値をセットされねばならない。</li> </ul>
01	<ul style="list-style-type: none"> <li>この状態は、PlayList の CPI_type が EP_map type である場合にだけ許される。</li> <li>先行する PlayItem と現在の PlayItem は、システムタイムベース（STC ベース）の不連続点があるために分割されていることを表す。</li> </ul>
10	<ul style="list-style-type: none"> <li>この状態は、PlayList の CPI_type が EP_map type である場合にだけ許される。</li> <li>この状態は、Virtual PlayList に対してだけ許される。</li> <li>先行する PlayItem と現在の PlayItem との接続は、シームレス再生の保証がなされている。</li> <li>先行する PlayItem と現在の PlayItem は、BridgeSequence を使用して接続されており、DVR MPEG-2 トランスポートストリームは、後述する DVR-STD に従っていなければならない。</li> </ul>
11	<ul style="list-style-type: none"> <li>この状態は、PlayList の CPI_type が EP_map type である場合にだけ許される。</li> <li>先行する PlayItem と現在の PlayItem は、シームレス再生の保証がなされている。</li> <li>先行する PlayItem と現在の PlayItem は、BridgeSequence を使用しないで接続されており、DVR MPEG-2 トランスポートストリームは、後述する DVR-STD に従っていなければならない。</li> </ul>

connection\_condition

【図 3 6】



【図 3 7】

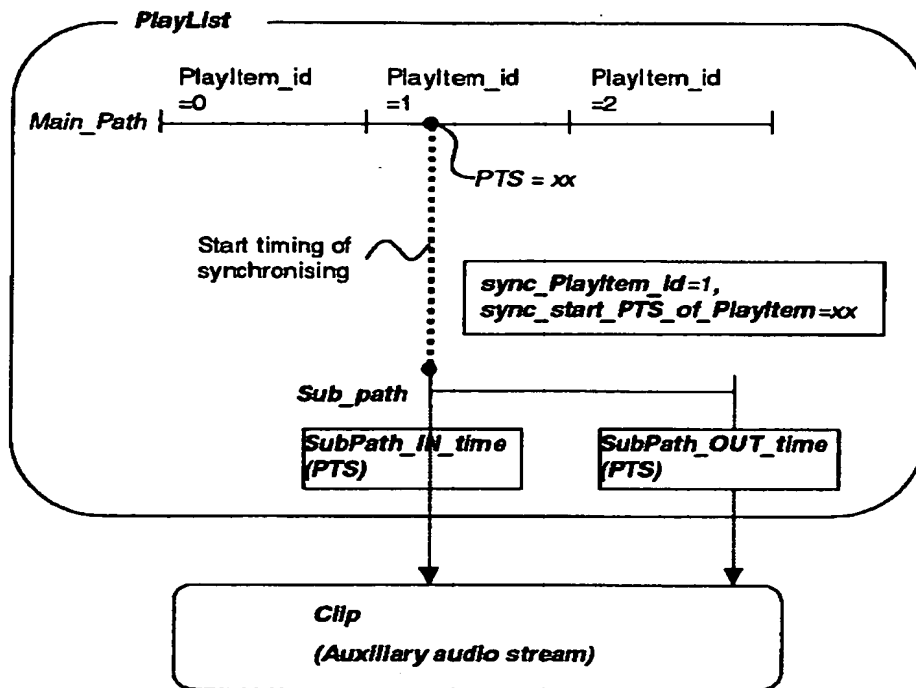


【図 3 8】

Syntax	No. of bits	Mnemonics
BridgeSequenceInfo() {		
Bridge_Clip_Information_file_name	8*10	bslbf
RSPN_exit_from_previous_Clip	32	uimsbf
RSPN_enter_to_current_Clip	32	uimsbf
}		

BridgeSequenceInfo のシンタックス

【図 3 9】



【図 4 0】

Syntax	No. of bits	Mnemonics
SubPlayItem() {		
Clip_Information_file_name	8*10	bslbf
SubPath_type	8	bslbf
sync_PlayItem_Id	8	uimsbf
sync_start_PTS_of_PlayItem	32	uimsbf
SubPath_IN_time	32	uimsbf
SubPath_OUT_time	32	uimsbf
}		

SubPlayItem のシンタックス

【図 4 1】

SubPath_type	Meaning
0x00	Auxiliary audio stream path
0x01 - 0xff	reserved

SubPath\_type

【図 4 2】

Syntax	No. of bits	Mnemonics
PlayListMark() {		
version_number	8*4	bslbf
length	32	uimsbf
number of PlayList marks	16	uimsbf
for( $l=0; l < \text{number of PlayList marks}; l++$ ) {		
reserved	8	bslbf
mark_type	8	bslbf
mark_time_stamp	32	uimsbf
PlayItem_id	8	uimsbf
reserved	24	uimsbf
character_set	8	bslbf
name_length	8	uimsbf
mark_name	8*256	bslbf
ref_thumbnail_index	16	uimsbf
}		
}		

PlayListMark のシンタックス



【図 4 3】

Mark type	Meaning	Comments
0x00	resume-mark	再生リジュームポイント。PlayListMark()において定義される再生リジュームポイントの数は、0 または 1 でなければならない。
0x01	book-mark	PlayList の再生エントリポイント。このマークは、ユーザがセットすることができ、例えば、お気に入り のシーンの開始点を指定するマークに使う。
0x02	skip-mark	スキップマークポイント。このポイントからプログラムの最後まで、プレーヤーはプログラムをスキップする。PlayListMark() において定義されるスキップ マークポイントの数は、0 または 1 でなければならない。
0x03 - 0x8F	reserved	
0x90 - 0xFF	reserved	Reserved for ClipMark()

mark\_type

【図 4 4】

CPI_type in the PlayList()	Semantics of mark_time_stamp
EP_map type	mark_time_stamp は、マークで参照されるプレゼンテーションユニットに対応する 33 ビット長の PTS の上位 32 ビットを示さなければならない。
TU_map type	mark_time_stamp は、TU_map_time_axis 上の時刻でなければならない。かつ、mark_time_stamp は、time_unit の精度に丸めて表さねばならない。mark_time_stamp は、次に示す等式により計算される。  $\text{mark\_time\_stamp} = \text{TU\_start\_time} \% 2^{32}$

mark\_time\_stamp

【図 4 5】

Syntax	No. of bits	Mnemonics
<b>zzzzz.cpi {</b>		
<b>STC Info Start address</b>	32	uimsbf
<b>ProgramInfo Start address</b>	32	uimsbf
<b>CPI Start address</b>	32	uimsbf
<b>ClipMark Start address</b>	32	uimsbf
<b>MakerPrivateData Start address</b>	32	uimsbf
<b>reserved</b>	96	bslbf
<b>ClipInfo()</b>		
for(i=0; i<N1; i++){		
padding word	16	bslbf
}		
<b>STC Info()</b>		
for(i=0; i<N2; i++){		
padding word	16	bslbf
}		
<b>ProgramInfo()</b>		
for(i=0; i<N3; i++){		
padding word	16	bslbf
}		
<b>CPI()</b>		
for(i=0; i<N4; i++){		
padding word	16	bslbf
}		
<b>ClipMark()</b>		
for(i=0; i<N5; i++){		
padding word	16	bslbf
}		
<b>MakerPrivateData()</b>		
<b>}</b>		

zzzzz.cpi のシンタクス

【図 4 6】

Syntax	No. of bits	Mnemonics
ClipInfo() {		
version number	8*4	bslbf
length	32	uimsbf
Clip stream type	8	bslbf
offset SPN	32	uimsbf
TS recording rate	24	uimsbf
reserved	8	bslbf
record time and date	4*14	bslbf
reserved	8	bslbf
duration	4*6	bslbf
reserved	7	bslbf
time controlled flag	1	bslbf
TS average rate	24	uimsbf
if (Clip stream type==1) // Bridge-Clip AV stream		
RSPN arrival time discontinuity	32	uimsbf
else		
reserved	32	bslbf
reserved for system use	144	bslbf
reserved	11	bslbf
is format identifier valid	1	bslbf
is original network ID valid	1	bslbf
is transport stream ID valid	1	bslbf
is service ID valid	1	bslbf
is country code valid	1	bslbf
format identifier	32	bslbf
original network ID	16	uimsbf
transport stream ID	16	uimsbf
service ID	16	uimsbf
country code	24	bslbf
stream format name	16*8	bslbf
reserved for future use	256	bslbf
}		

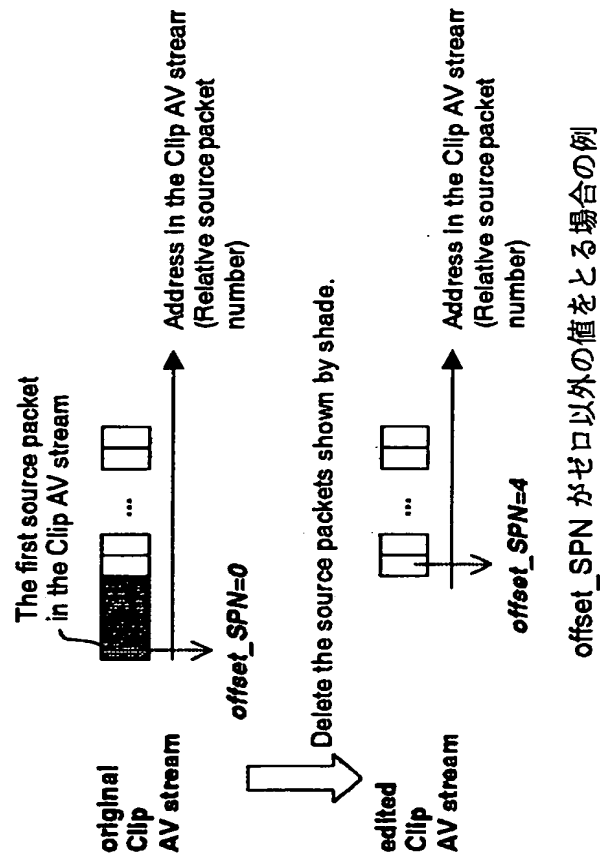
## ClipInfo のシンタクス

【図 4 7】

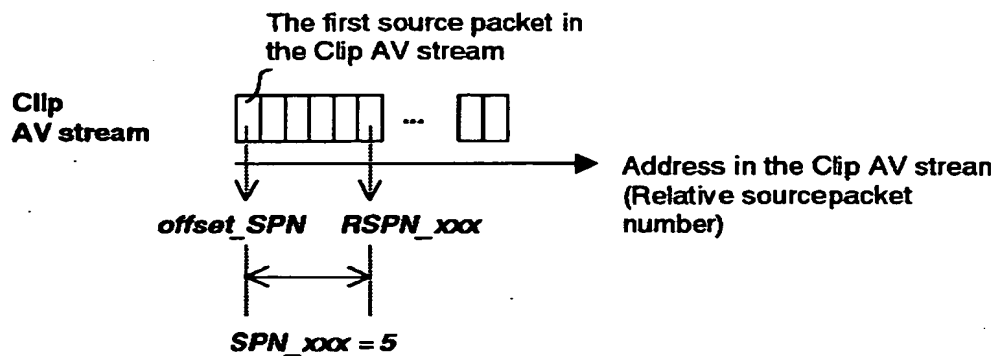
Clip_stream_type	meaning
0	Clip AV ストリーム
1	Bridge-Clip AV ストリーム
2 - 255	Reserved

Clip\_stream\_type

【図 4 8】



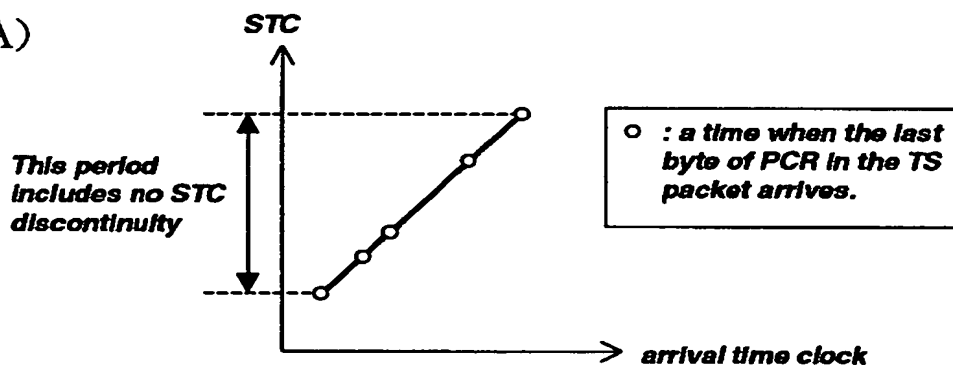
【図 4 9】



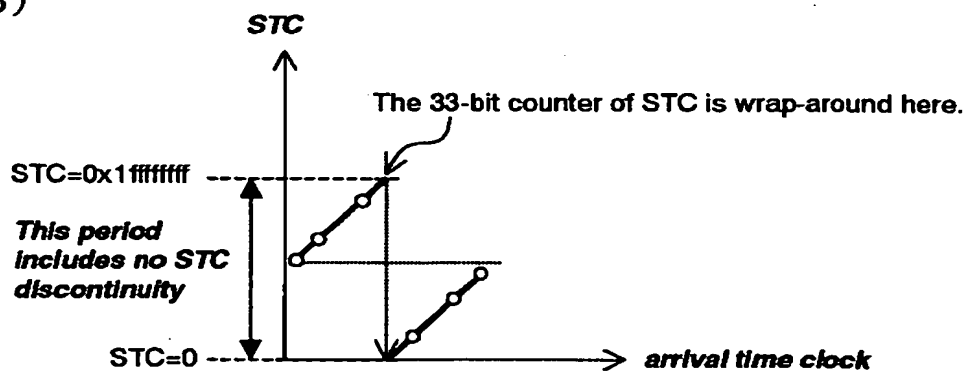
AV ストリームでの *offset\_SPN* と相対ソースパケット番号 (*RSPN\_xxx*) の間の関係

【図 5 0】

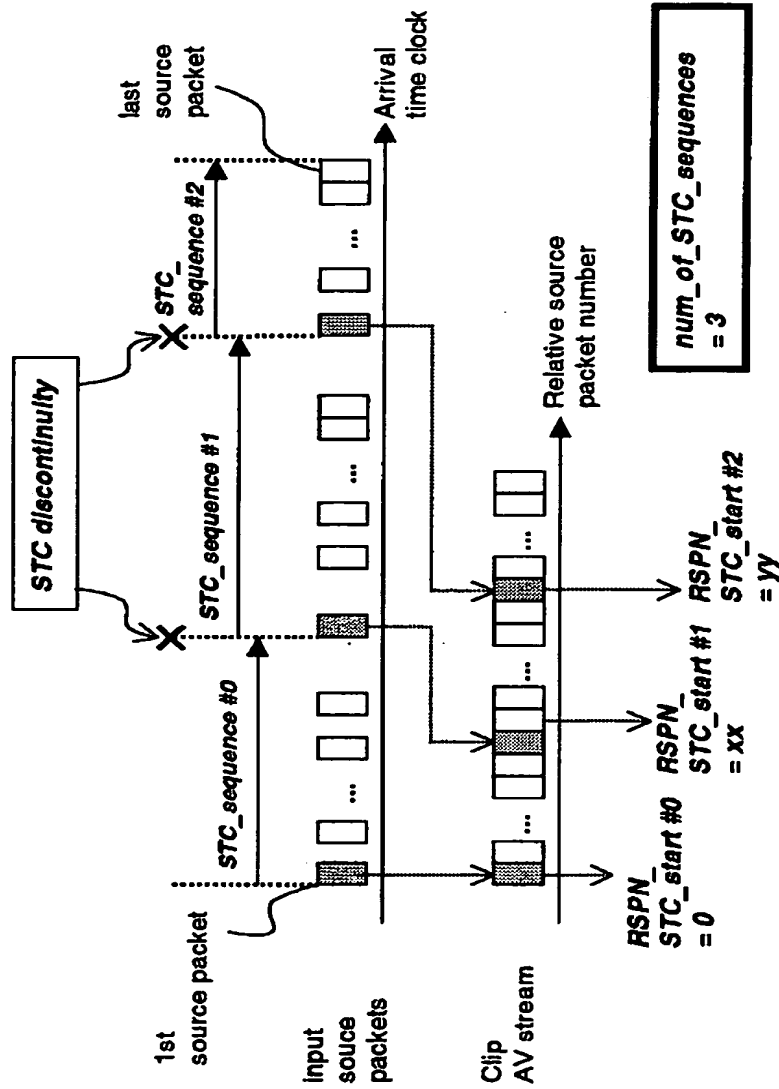
(A)



(B)



【図 51】



--STC\_Info

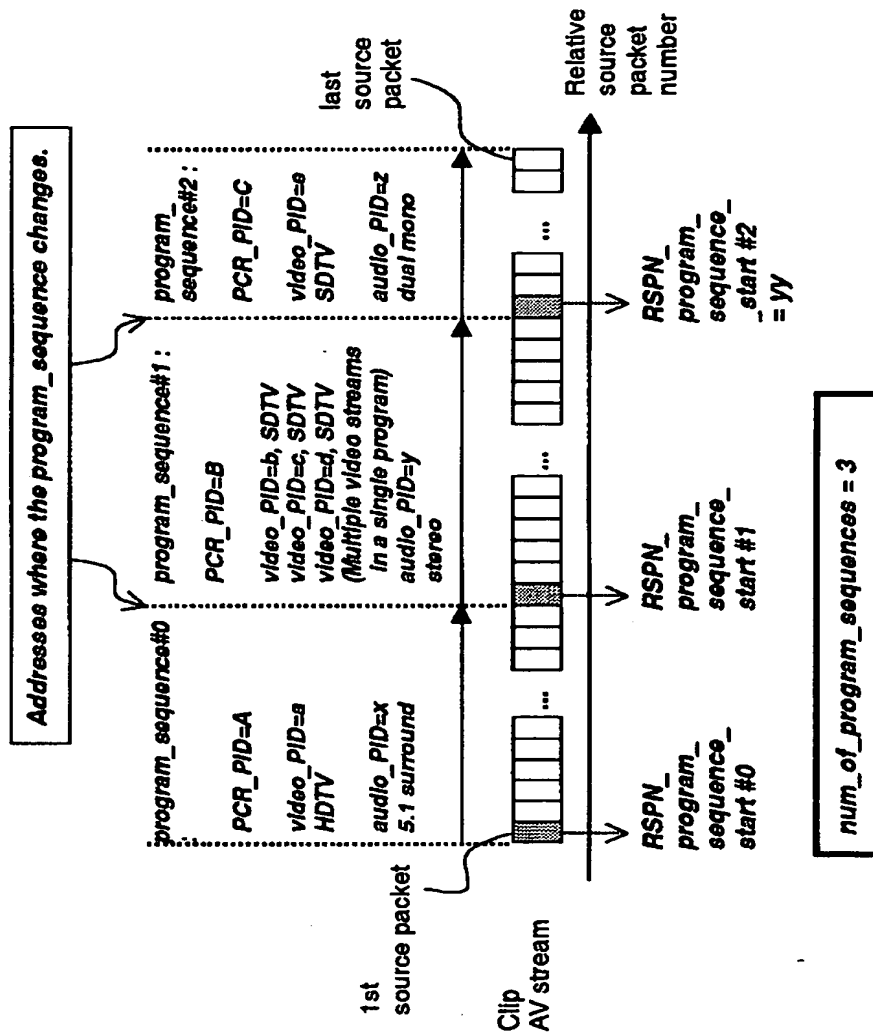


【図 5 2】

Syntax	No. of bits	Mnemonics
STC_Info() {		
version_number	8*4	bslbf
length	32	uimbsf
if (length != 0) {		
reserved	8	bslbf
num_of_STC_sequences	8	uimbsf
for(STC_sequence_id=0;		
STC_sequence_id < num_of_STC_sequences;		
STC_sequence_id++) {		
reserved	32	bslbf
RSPN_STC_start	32	uimbsf
}		
}		
}		

STC\_Infoのシンタックス

【図 53】



ProgramInfo の例

【図 5 4】

Syntax	No. of bits	Mnemonics
ProgramInfo() {		
version_number	8*4	bslbf
length	32	uimsbf
if (length != 0) {		
reserved	8	bslbf
number_of_program_sequences	8	uimsbf
for(i=0; i<number_of_program_sequences; i++){		
RSPN_program_sequence_start	32	uimsbf
reserved	48	bslbf
PCR_PID	16	bslbf
number_of_videos	8	uimsbf
number_of_audios	8	uimsbf
for (k=0; k<number_of_videos; k++) {		
video_stream_PID	16	bslbf
VideoCodingInfo()		
}		
for (k=0; k<number_of_audios; k++) {		
audio_stream_PID	16	bslbf
AudioCodingInfo()		
}		
}		
}		

ProgramInfo のシンタックス

【図 5 5】

Syntax	No. of bits	Mnemonics
VideoCodingInfo() {		
video_format	8	uimsbf
frame_rate	8	uimsbf
display_aspect_ratio	8	uimsbf
reserved	8	bslbf
}		

VideoCodingInfo のシンタックス

【図 5 6】

video_format	Meaning
0	480i
1	576i
2	480p (including 640x480p format)
3	1080i
4	720p
5	1080p
6 - 254	reserved
255	No information

video\_format

【図 5 7】

frame_rate	Meaning
0	forbidden
1	24 000/1001 (23.976...)
2	24
3	25
4	30 000/1001 (29.97..)
5	30
6	50
7	60 000/1001 (59.94 ..)
8	60
9 - 254	reserved
255	No information

frame\_rate

【図 5 8】

display_aspect_ratio	Meaning
0	forbidden
1	reserved
2	4:3 display aspect ratio
3	16:9 display aspect ratio
4-254	reserved
255	No information

display\_aspect\_ratio

【図 5 9】

Syntax	No. of bits	Mnemonics
AudioCodingInfo() {		
audio_coding	8	uimsbf
audio_component_type	8	uimsbf
sampling_frequency	8	uimsbf
reserved	8	bslbf
}		

AudioCodingInfo のシンタックス

【図 6 0】

audio_coding	Meaning
0	MPEG-1 audio layer I or II
1	Dolby AC-3 audio
2	MPEG-2 AAC
3	MPEG-2 multi-channel audio, backward compatible to MPEG-1
4	SESF LPCM audio
5-254	reserved
255	No information

audio\_coding



【図 6 1】

audio_component_type	Meaning
0	single mono channel
1	dual mono channel
2	stereo (2-channel)
3	multi-lingual, multi-channel
4	surround sound
5	audio description for the visually impaired
6	audio for the hard of hearing
7-254	reserved
255	No information

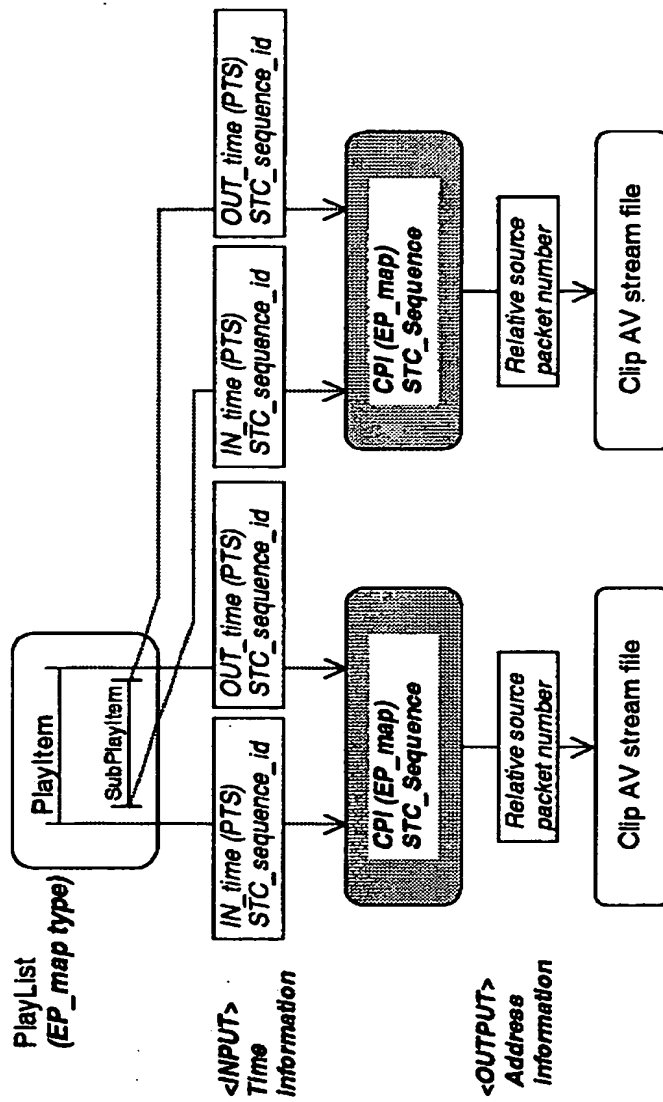
audio\_component\_type

【図 6 2】

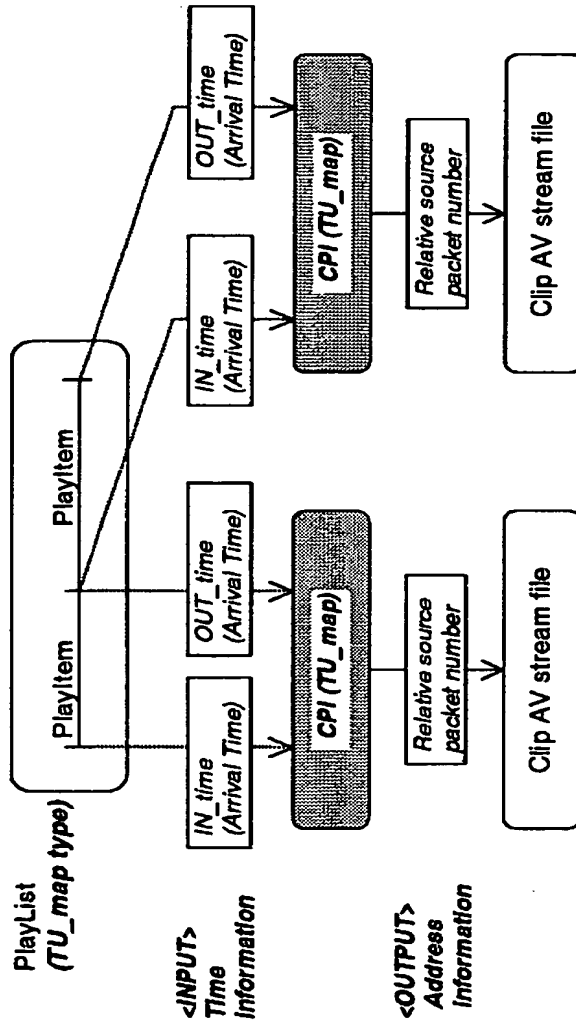
sampling_frequency	Meaning
0	48 kHz
1	44.1 kHz
2	32 kHz
3-254	reserved
255	No information

sampling\_frequency

【図 63】



【図 6 4】



【図 65】

Syntax	No. of bits	Mnemonics
CPI0 {		
version_number	8*4	bslbf
length	32	uimsbf
reserved	15	bslbf
CPI_type	1	bslbf
if (CPI_type == 0)		
EP_map0		
else		
TU_map0		
}		

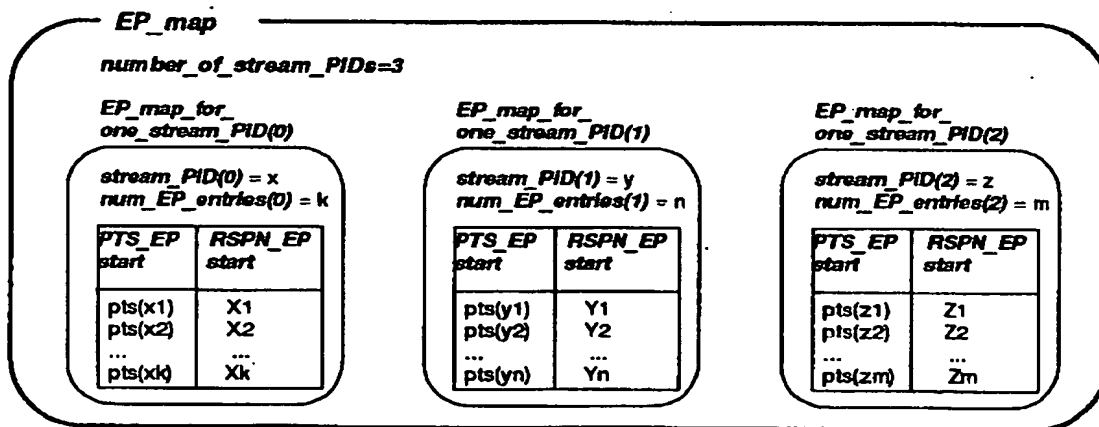
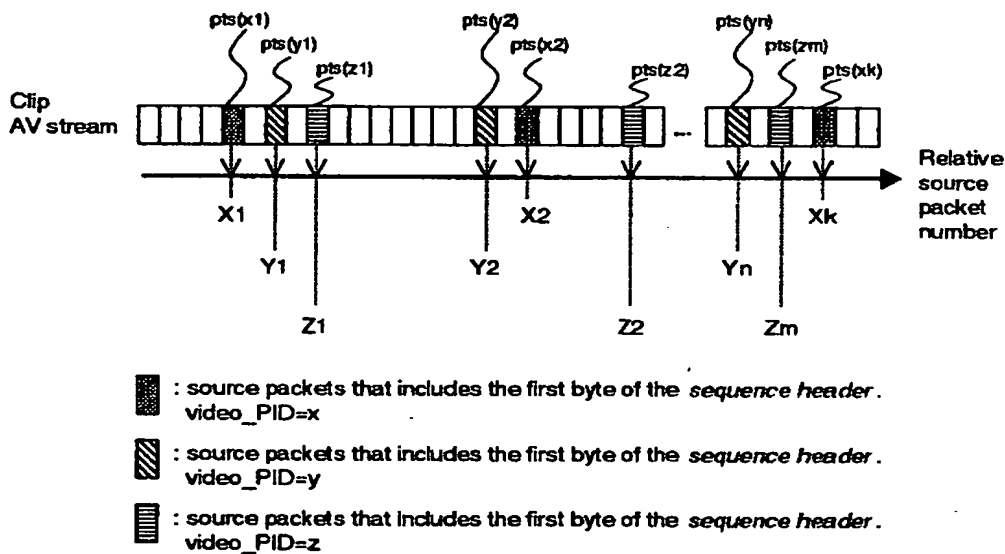
CPI のシンタックス

【図 66】

CPI_type	Meaning
0	EP map type
1	TU map type

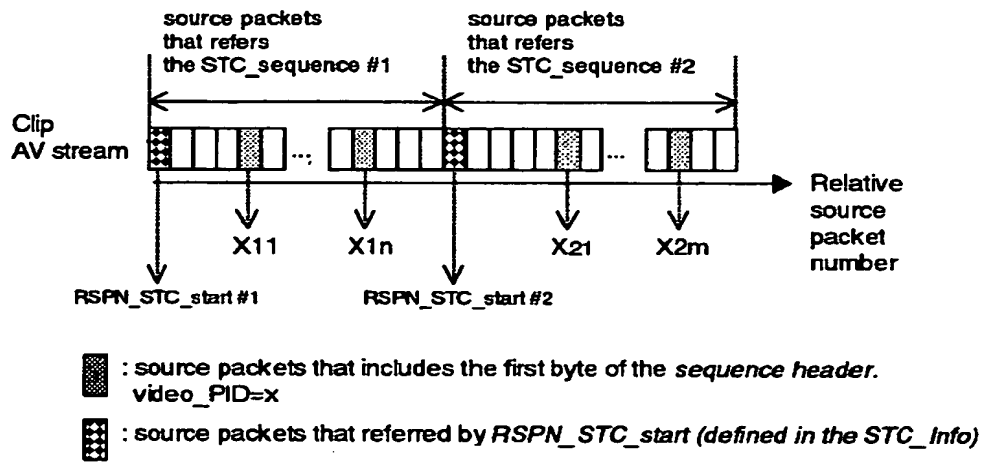
### CPI\_type の意味

【図 67】



### ビデオの EP\_map の例

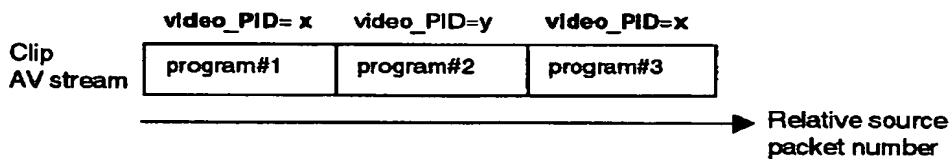
【図 6 8】

**EP\_map\_for\_one\_stream\_PID***video\_PID=x*

<i>PTS_EP start</i>	<i>RSPN_EP start</i>	
pts(x11)	X11	These data belong to the STC_sequence #1
...	...	
pts(x1n)	X1n	
		→ boundary
pts(x21)	X21	These data belong to the STC_sequence #2
...	...	
pts(x2m)	X2m	

**RSPN\_STC\_start #2 < X21**

【図 6 9】

**EP\_map***number\_of\_stream\_PIDs=3**EP\_map\_for one\_stream\_PID(0)**stream\_PID(0) = x*  
...*EP\_map\_for one\_stream\_PID(1)**stream\_PID(1) = y*  
...*EP\_map\_for one\_stream\_PID(2)**stream\_PID(2) = x*  
...

【図 7 0】

Syntax	Bits	Mnemonics
Thumbnail() {		
version_number	8*4	char
length	32	uimsbf
if (length != 0) {		
tn_blocks_start_address	32	bslbf
number_of_thumbnails	16	uimsbf
tn_block_size	16	uimsbf
number_of_tn_blocks	16	uimsbf
reserved	16	bslbf
for(l = 0; l < number_of_thumbnails; l++) {		
thumbnail_index	16	uimsbf
thumbnail_picture_format	8	bslbf
reserved	8	bslbf
picture_data_size	32	uimsbf
start_tn_block_number	16	uimsbf
x_picture_length	16	uimsbf
y_picture_length	16	uimsbf
reserved	16	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for(k = 0; k < number_of_tn_blocks; k++) {		
tn_block	tn_block_size* 1024*8	
}		
}		

Thumbnail シンタックス

【図 7 1】

EP_type	Meaning
0	video
1	audio
2 - 15	reserved

EP\_type Values

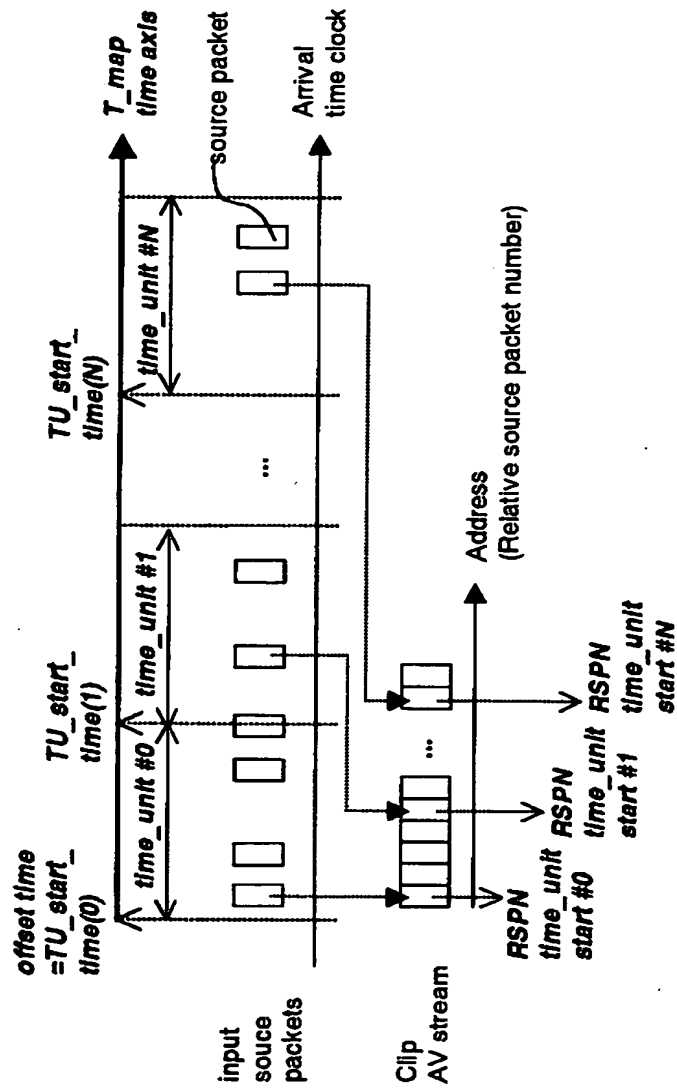


【図 72】

Syntax	No. of bits	Mnemonics
EP_map_for_one_stream_PID(N){		
for (l=0; l< N; l++) {		
PTS_EP_start	32	uimsbf
RSPN_EP_start	32	uimsbf
}		
}		

EP\_map\_for\_one\_stream\_PID のシンタックス

【図 73】



【図 7 4】

Syntax	No. of bits	Mnemonics
TU_map0{		
offset_time	32	bslbf
time_unit_size	32	uimsbf
number_of_time_unit_entries	32	uimsbf
for (k=0; k<number_of_time_unit_entries; k++)		
RSPN_time_unit_start	32	uimsbf
}		

TU\_mapのシンタックス

【図 7 5】

Syntax	No. of bits	Mnemonics
ClipMark() {		
version_number	8*4	bslbf
length	32	uimsbf
number of Clip_marks	16	uimsbf
for(=0; i < number of Clip_marks; i++) {		
reserved	8	bslbf
mark_type	8	bslbf
mark_time_stamp	32	uimsbf
STC_sequence_id	8	uimsbf
reserved	24	bslbf
character_set	8	bslbf
name_length	8	uimsbf
mark_name	8*256	bslbf
ref_thumbnail_index	16	uimsbf
}		
}		

ClipMark のシンタックス

【図 7 6】

Mark type	Meaning	Comments
0x00 - 0x8F	reserved	Reserved for PlayListMark()
0x90	Event-start mark	番組の開始ポイントを示すマーク点。
0x91	Local event-start mark	番組の中の局所的な場面を示すマーク点。
0x92	Scene-start mark	シーンチェンジポイントを示すマーク。
0x93 - 0xFF	reserved	

mark\_type

【図 7 7】

CPI_type in the CPIQ	Semantics of mark_time_stamp
EP_map type	mark_time_stamp は、マークで参照されるプレゼンテーションユニットに対応する 33 ビット長の PTS の上位 32 ビットを示さなければならぬ。
TU_map type	mark_time_stamp は、TU_map_time_axis 上の時刻でなければならぬ。かつ、mark_time_stamp は、time_unit の精度に丸めて表さねばならぬ。mark_time_stamp は、次に示す等式により計算される。  $\text{mark\_time\_stamp} = \text{TU\_start\_time} \% 2^{32}$

mark\_type\_stamp

【図 7 8】

Syntax	No. of bits	Mnemonics
menu.thmb / mark.thmb {		
reserved	256	bslbf
Thumbnail()		
for(i=0; i<N1; i++)		
padding_word	16	bslbf
}		

menu.thmb と mark.thmb のシンタックス

【図 7 9】

Syntax	Bits	Mnemonics
Thumbnail() {		
version_number	8*4	char
length	32	uimsbf
if (length != 0) {		
tn_blocks_start_address	32	bslbf
number_of_thumbnails	16	uimsbf
tn_block_size	16	uimsbf
number_of_tn_blocks	16	uimsbf
reserved	16	bslbf
for (l = 0; l < number_of_thumbnails; l++) {		
thumbnail_index	16	uimsbf
thumbnail_picture_format	8	bslbf
reserved	8	bslbf
picture_data_size	32	uimsbf
start_tn_block_number	16	uimsbf
x_picture_length	16	uimsbf
y_picture_length	16	uimsbf
reserved	16	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for (k = 0; k < number_of_tn_blocks; k++) {		
tn_block	tn_block_size* 1024*8	
}		
}		

Thumbnailのシンタックス

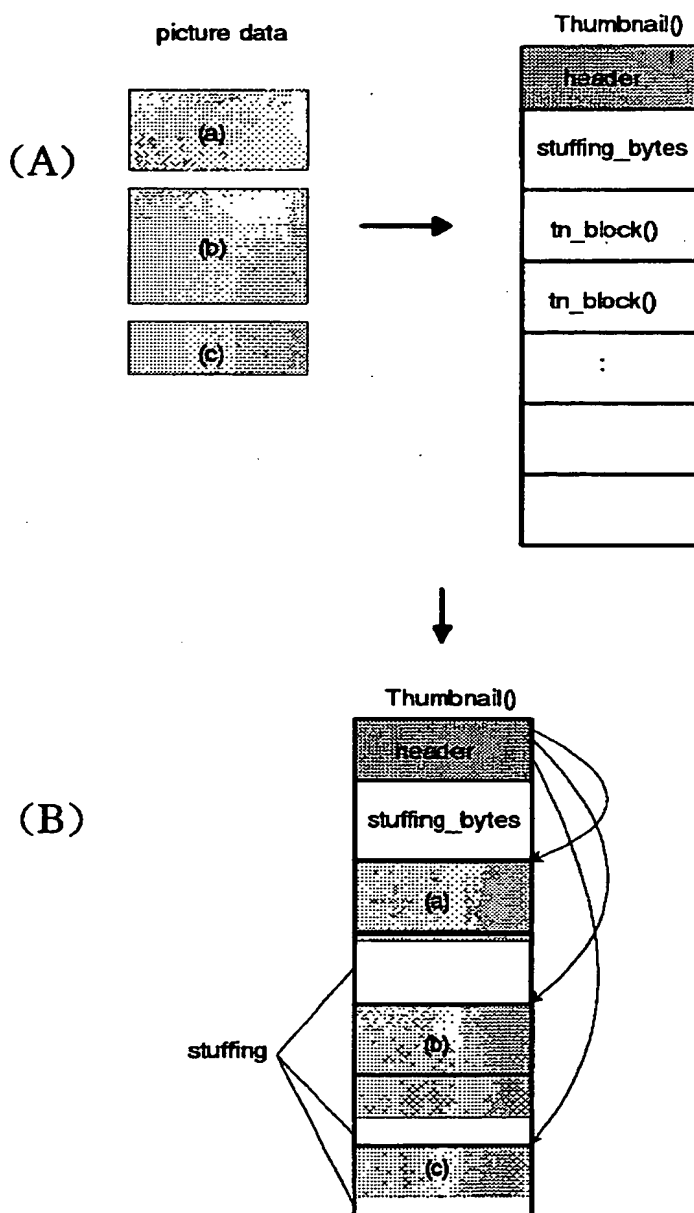


【図 8 0】

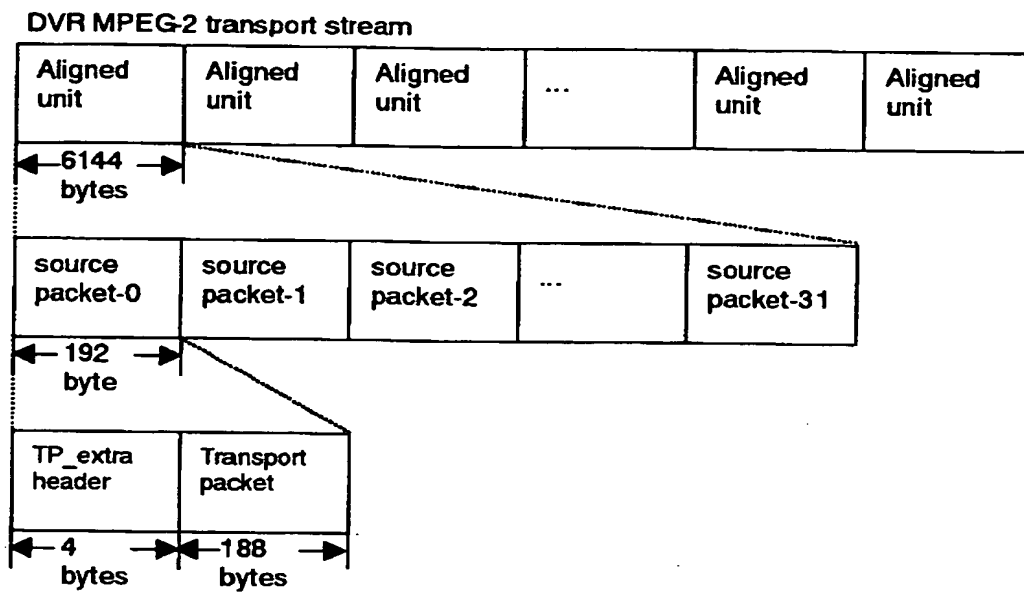
<i>Thumbnail picture format</i>	<i>Meaning</i>
<i>0x00</i>	<i>MPEG-2 Video I-picture</i>
<i>0x01</i>	<i>DCF (restricted JPEG)</i>
<i>0x02</i>	<i>PNG</i>
<i>0x03-0xff</i>	<i>reserved</i>

thumbnail\_picture\_format

【図 8 1】

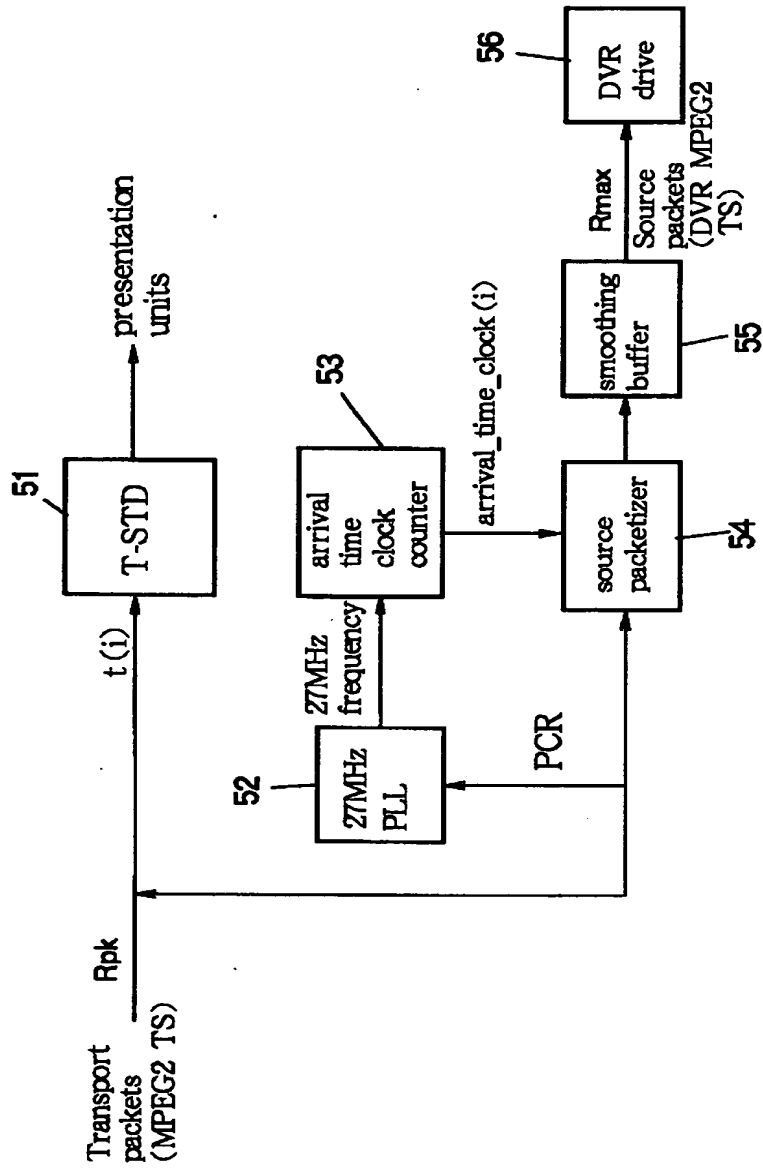


【図 8 2】



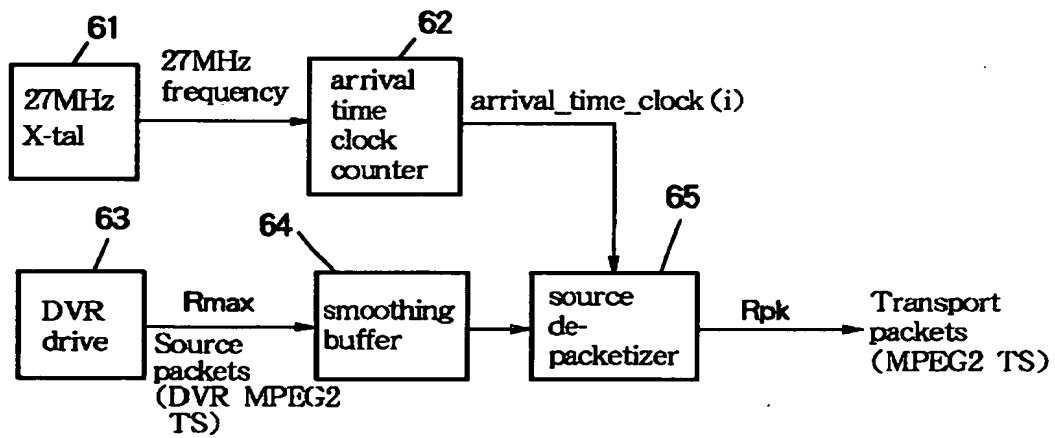
DVR MPEG-2 トランスポートストリームの構造

【図 83】



DVR MPEG-2 トランスポートストリームのレコーダモデル

【図 84】



DVR MPEG-2 トランスポートストリームのプレーヤモデル

【図 8 5】

Syntax	No. of bits	Mnemonics
source_packet () {		
TP_extra_header()		
transport_packet()		
}		

source packet

【図 8 6】

Syntax	No. of bits	Mnemonics
TP_extra_header() {		
copy_permission_indicator	2	uimsbf
arrival_time_stamp	30	uimsbf
}		

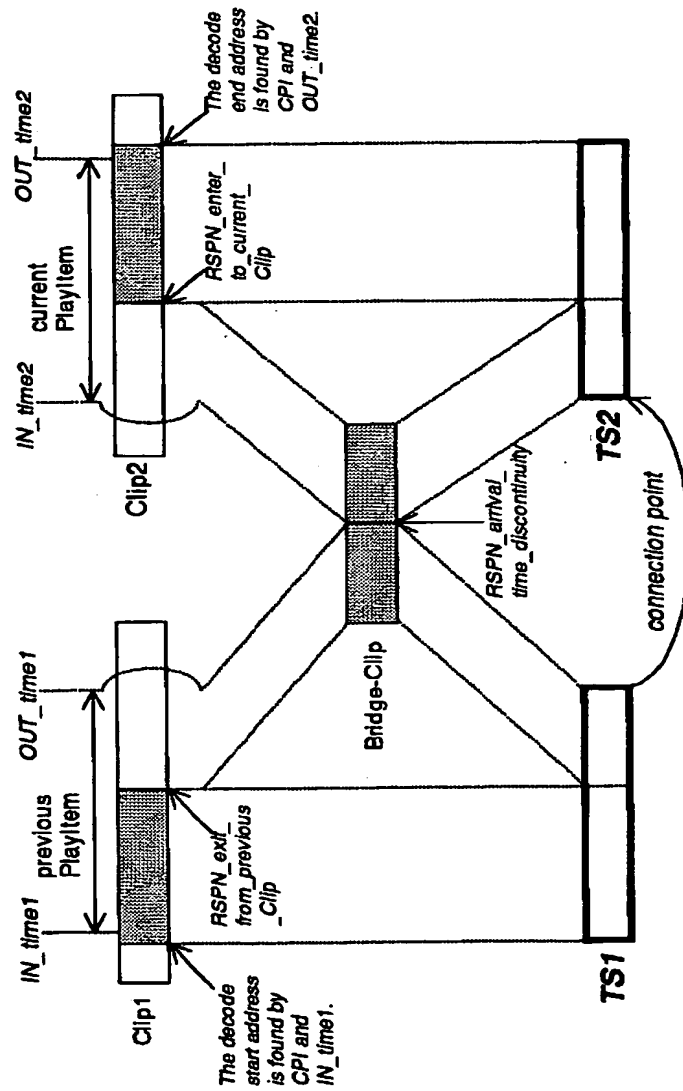
TP\_extra\_header

【図 8 7】

copy_permission indicator	meaning
00	copy free
01	no more copy
10	copy once
11	copy prohibited

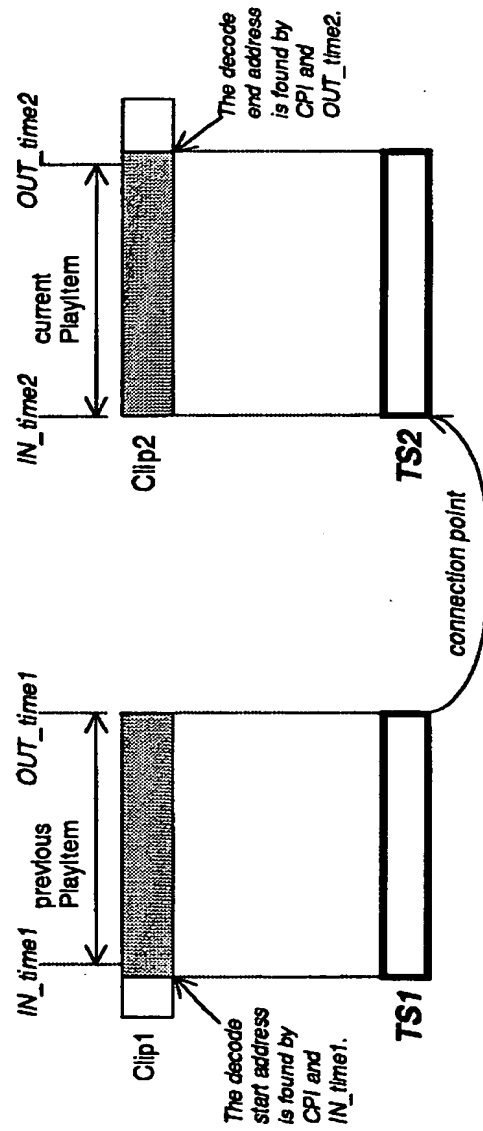
copy permission indicator table

【図 88】

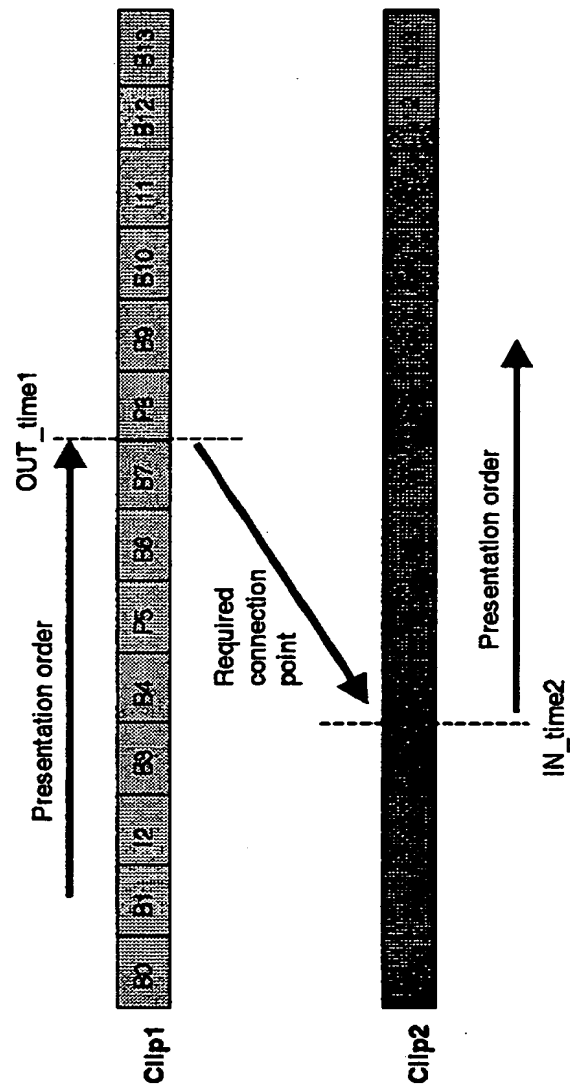




【図 89】

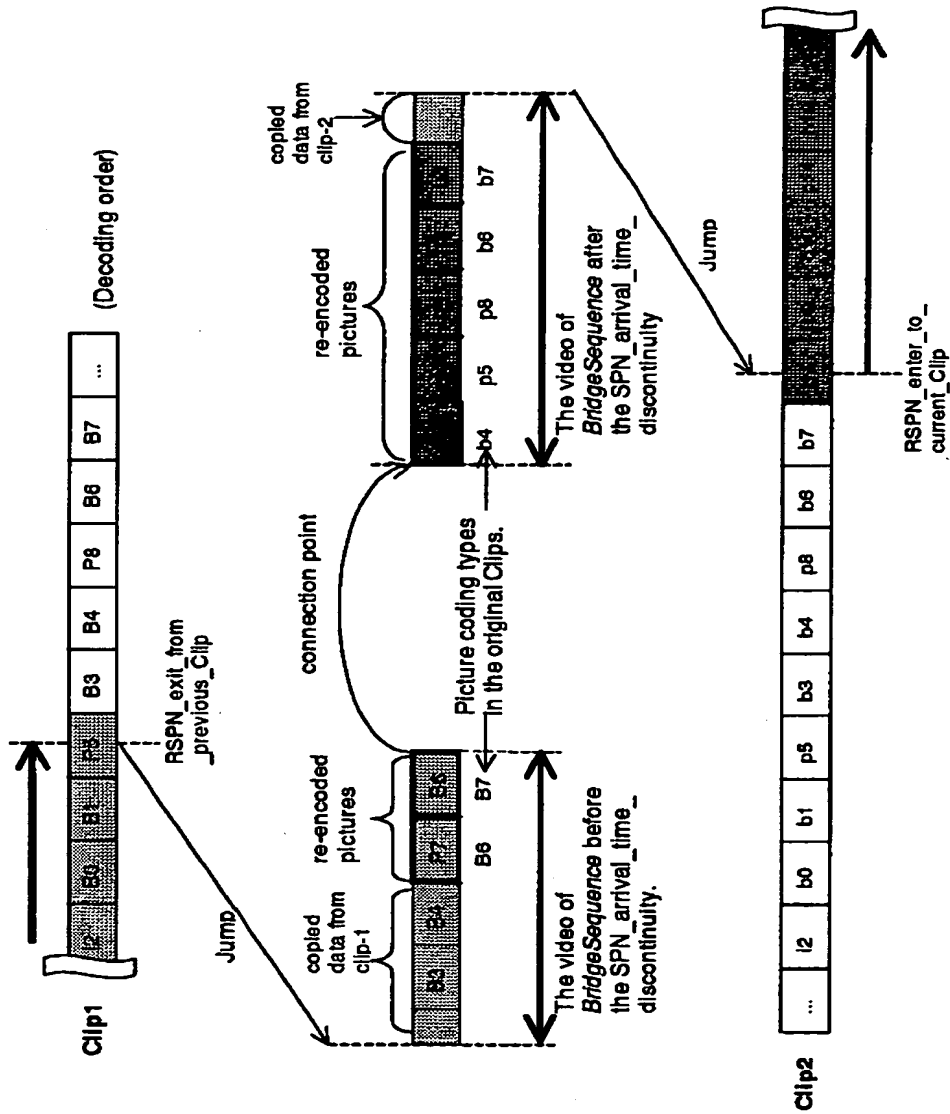


【図 9 0】



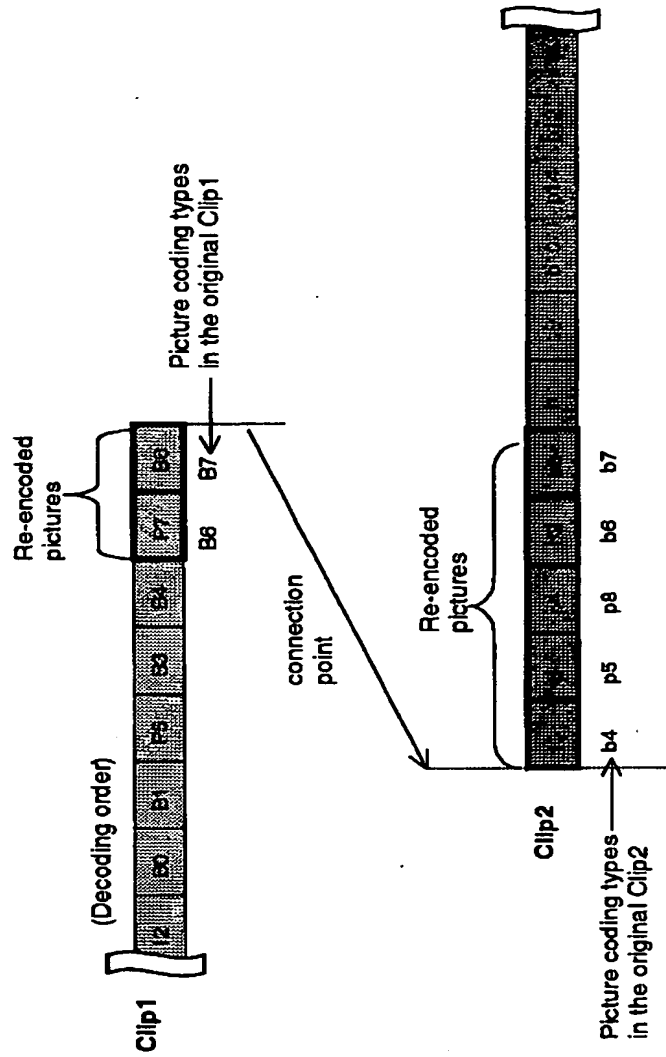
ビクチャの表示順序で示すシームレス接続の例

【図 91】



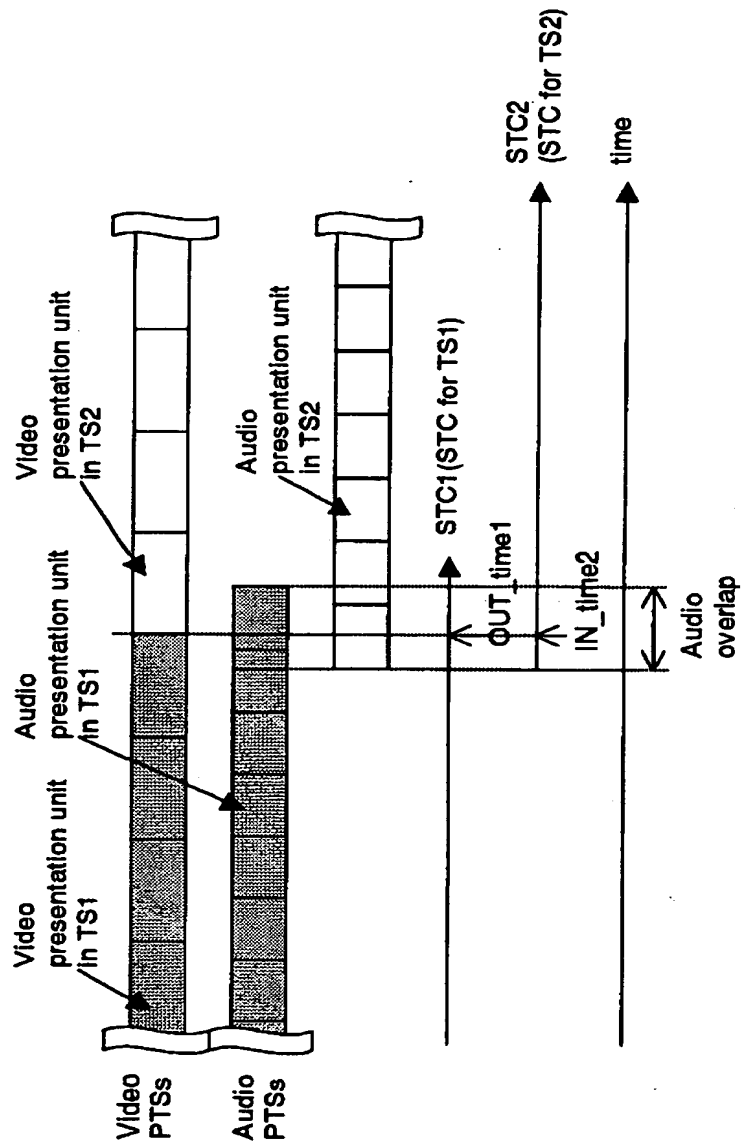
BridgeSequence を使用してシームレス接続を実現する例 1

【図 9 2】

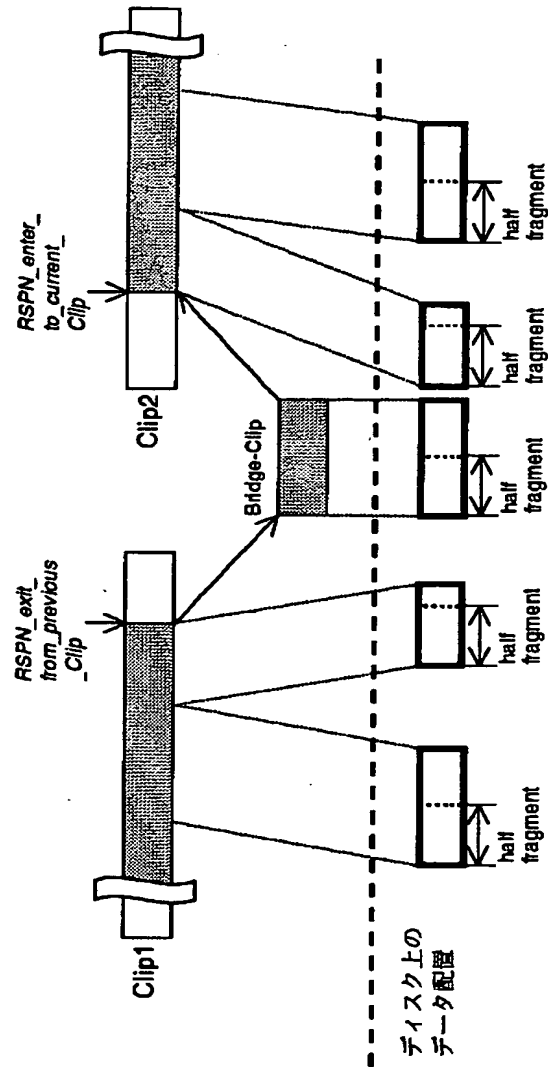


BridgeSequence を使用しないでシームレス接続を実現する例 2

【図93】

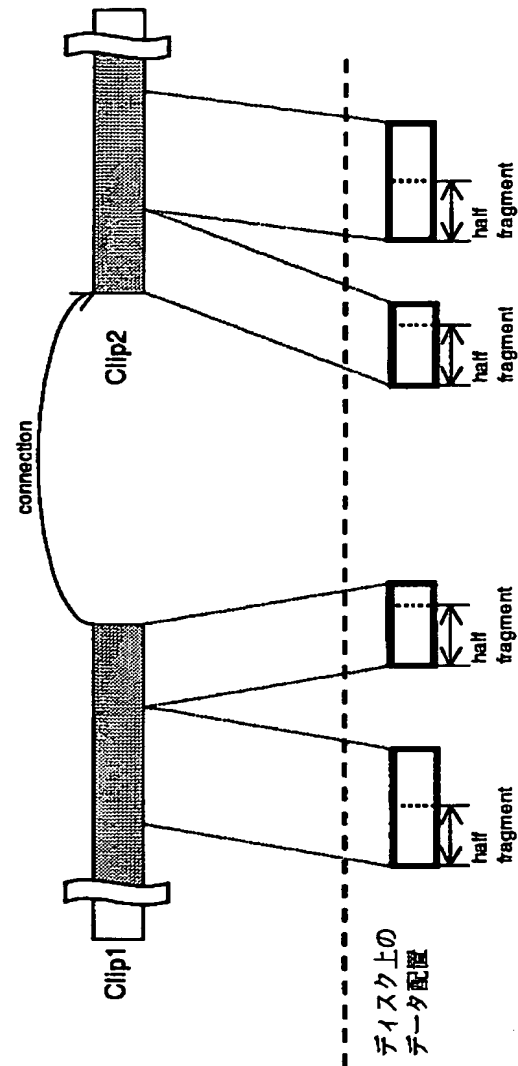


【図94】



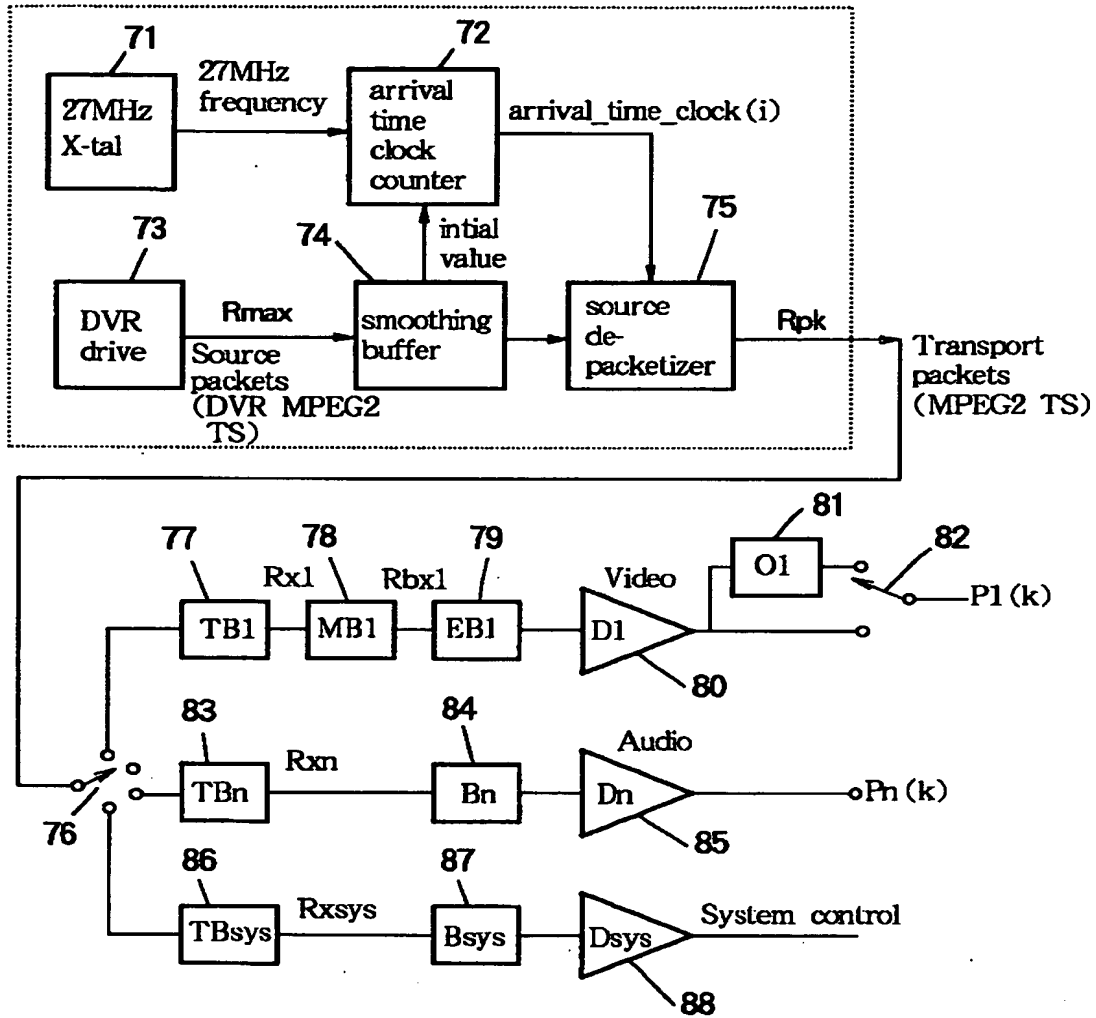
BridgeSequence を使用してシームレス接続をする場合の、データアロケーションの例

【図 9 5】



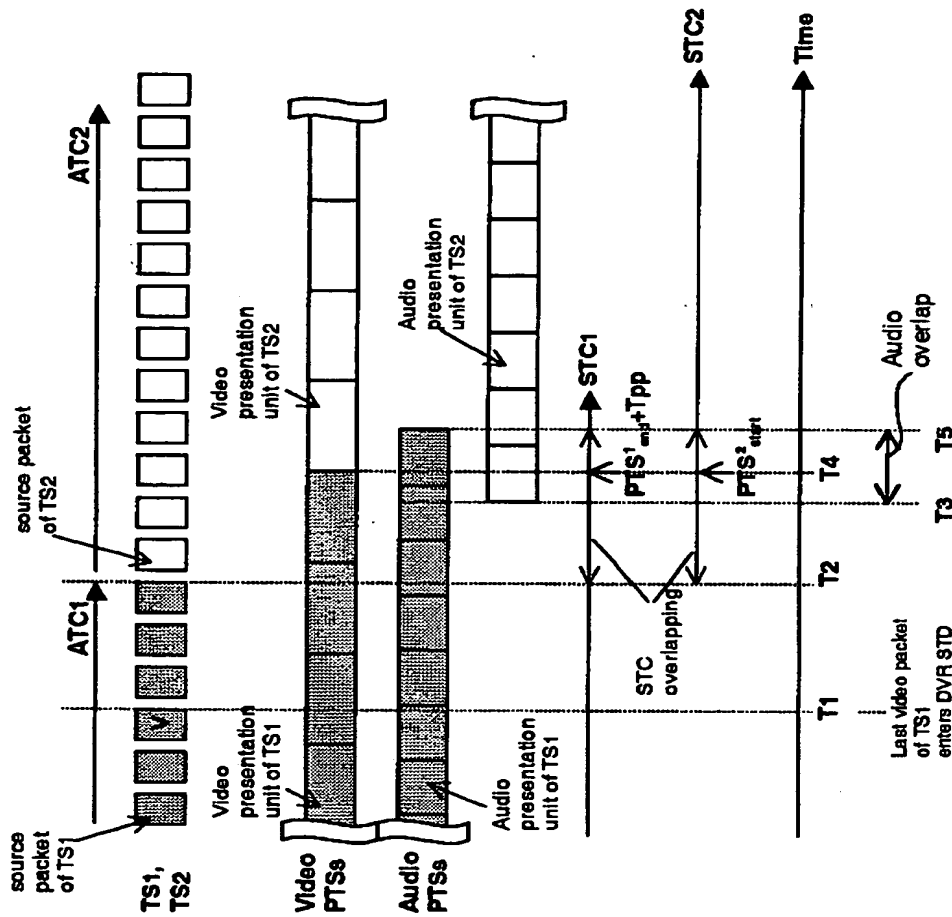
BridgeSequence を使用しないでシームレス接続をする場合の、データアロケーションの例

【図 96】



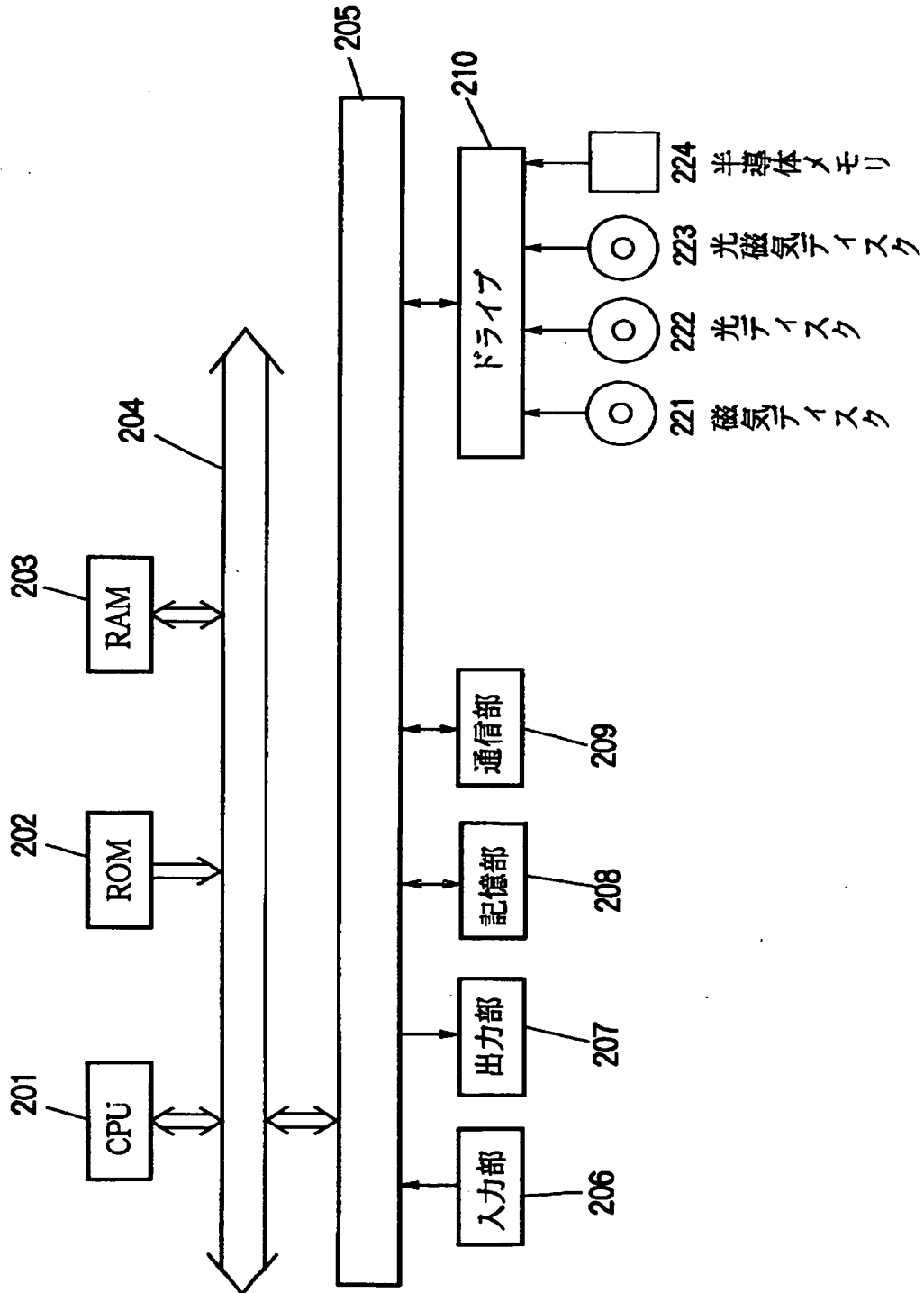


【図 97】



ある AV ストリーム (TS1) からそれにシームレスに接続された次の AV ストリーム (TS2) へと移る時のトランスポートパケットの入力、復号、表示のタイミングチャート

【図 98】



【書類名】 要約書

【要約】

【課題】 別々に記録された動画像の連続性を保つように再生できるようにする

。 【解決手段】 別々に記録されたClip1とClip2を連続再生するとき、Clip1からClip2へと橋渡しの役割をもつBridge Clipが生成される。Bridge Clipは、Clip1からClip2へと切り替わる部分の、Clip1とClip2との、それぞれ対応する部分から構成される。

【選択図】 図37

出 願 人 履 歴 情 報

識別番号 [000002185]

1. 変更年月日 1990年 8月30日

[変更理由] 新規登録

住 所 東京都品川区北品川6丁目7番35号

氏 名 ソニー株式会社